

ASSESSMENT AND INTERNAL VERIFICATION FRONT SHEET (Individual Criteria)

Course Title	Bachelor of Science (Honours) in Software Development			Lecturer Name & Surname	James Attard	
Unit Number & Title		ITSFT-506-1612 Server Side Scripting				
Assignment Number, Title / Type		Home Assignment				
Date Set		20 th December, 2023	Deadline Date	15 th January, 2023		
Student Name			ID Number		Class / Group	

Assessment Criteria	Maximum Mark
AA1.3: Construct a program to show the difference between client side and server side scripting	7
KU1.4: Show understanding of typical features in a framework	5
KU2.2: Show understanding of various helpers and components available in a framework	5
AA2.3: Develop a web application using the features of a server side framework	7
SE2.4: Develop and design an application programming interface using the features of server side framework	10
KU3.2: Show understanding of the validation techniques available in a framework	5
SE3.3: Design URLs in a SEO friendly way	10
AA4.2: Create table associations to be used in an application using the features of a server side framework	7
SE4.3: Modify a web application in order to make it safer using the framework's security features	10
Total Mark	66

Notes to Students:
<ul style="list-style-type: none"> This assignment brief has been approved and released by the Internal Verifier through Classter. Assessment marks and feedback by the lecturer will be available online via Classter (http://mcast.classter.com) following release by the Internal Verifier Students submitting their assignment on Moodle/Turnitin will be requested to confirm online the following statements: <ul style="list-style-type: none"> Student's declaration prior to handing-in of assignment <ul style="list-style-type: none"> ❖ I certify that the work submitted for this assignment is my own and that I have read and understood the respective Plagiarism Policy Student's declaration on assessment special arrangements <ul style="list-style-type: none"> ❖ I certify that adequate support was given to me during the assignment through the Institute and/or the Inclusive Education Unit. ❖ I declare that I refused the special support offered by the Institute.

Instructions

1. This is a home-based assessment comprising a project made up of multiple sub-tasks.
2. Any form of copying is prohibited. This includes plagiarism, AI tools, etc.
3. Late submissions will not be accepted and will be considered as not submitted.
4. The entire project must be version controlled on to a public repository in GitHub. Students must version control each task in to its own branch.
5. The deliverables of this assessment must include two files that are bundled into one single zip file. The two files inside this bundle are:
 - a. The codebase of the project zipped in .zip format.
 - b. A Microsoft Word document (.docx format) including:
 - i. The Cover Sheet filled with your details, as the first page of this document.
 - ii. Link to the GitHub repository where the entire project is located. The link must be publicly accessible. Ex. <https://www.github.com/joeborg/server-side-scripting-project-2023>
 - iii. Subtitles of each sub-task and references to the branch name. All the answers should be clearly labelled on which git branch is the code located. Ex. Task 1: "01-blade-layout-setup"; Task 2: "03-api-setup"; Task 3: "03-dynamic-data-reading", etc.
 - iv. A YouTube link to a screen recording session of your web application, demonstrating all the features required by this assessment. Video does not need to have any audio and must not be longer than two minutes in length.

Please refer to the Sample Assessment Format for more information.

6. The final document should be uploaded on VLE. Make sure to choose the correct link for your class.
7. The document should be named in the format class_surname_name (without the dots). For example, SWD62B_Borg_Joseph.
8. Interviews to demonstrate the project will be scheduled to the following week after the assignment deadline.

Sample Assessment Format

< Cover Sheet as First Page >

GitHub Repository

The code for this project is located on <https://github.com/jamesattard/serversidescripting-project>.

Task 1 – Database Setup

01-some-branch (replace with actual branch name)

02-you-can-have-multiple-branches-if-you-want

Task 2 – Template Layout

03-some-other-branch/s

Task 3 – Read Data

04-some-other-branch/s

Task 4 – Create Data

05-some-other-branch/s

Task 5 – Edit Data

06-some-other-branch/s

Task 6 – Delete Data

07-some-other-branch/s

...

Demo

A working demo of the web application can be found on <https://youtu.be/dQw4w9WgXcQ>

Carozza App Specifications

Carozza Ltd. is a new company based in Malta that stocks and sells various cars. The company has made deals with the following four car manufacturers and only intends to import cars solely from them:

1. Toyota Motor Corp
2. BMW Group
3. Audi AG
4. Honda Motor Corp

The company has hired you to create their online car stock system, Carozza App. They have written this requirement specification document, together with some screenshots of the UI to help you visualize the system. The system should be built in Laravel 10.x using an MVC approach.

A. Database Specification

Carozza App must have a database called **carozzaapp_db** and have the following two tables:

1. A table called **manufacturers**, that needs to be manually populated with the data of the above four suppliers. This table must have the following definition:

id (auto-generated)
name (varchar)
address (varchar) -> nullable()
phone (varchar) -> nullable()

Example data inside this table:

- 1 | Toyota Motor Corporation | HQ, Kyoto District, Tokyo, Japan | +1 800 233 8232
- 2 | BMW Group | HQ, Bavaria State, Berlin, Germany | +12 234 8552 923

2. A table called **cars**, that will be managed through the Carozza App web application. This table must have the following definition:

id (auto-generated)

model (varchar)
year (varchar)
salesperson_email (varchar)
manufacturer_id (foreign_id)

Example data inside this table:

1 | Camry | 2010 | joe@carozza.com | 1
2 | Hilux | 2020 | mary@carozza.com | 1
3 | 330i | 2021 | joe@carozza.com | 2

B. UI Specification

Carozza App must have an appropriate UI layout to make it easier to navigate while having a professional sleek look. You are required to use a Bootstrap based template that can integrate well with Laravel's Blade templating engine. This is how the application should look like when visiting the main page:

CAROZZA APP Manufacturers Cars

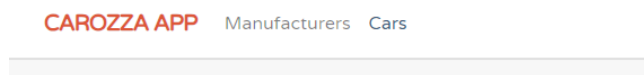
All Cars Models					+ Add New
All Manufacturers					
#	Model	Year	Salesperson Email	Manufacturer	Actions
1	Camry	2010	joe@carozza.com	Toyota Motor Corp	View Edit Delete
2	Hilux	2020	mary@carozza.com	Toyota Motor Corp	View Edit Delete
3	330i	2021	joe@carozza.com	BMW Group	View Edit Delete

You must also refer to the mockups inside the section Features Specification for the rest of the screen mockups.

C. Features Specification

C1. Navigation Bar

Carozza App must be accessed on <http://<your-ip-address>/cars> and should display a navigation bar that allows the user to either view a list of the manufacturers or the list of the cars in a tabular format:



Clicking on the manufacturers tab, should simply display the list of the manufacturers. This is accessible on <http://<your-ip-address>/manufacturers>:

All Manufacturers			
#	Name	Address	Phone
1	Toyota Motor Corporation	HQ, Kyoto District, Tokyo Japan	+1 800 233 8232
2	BMW GGroup	HQ, Bavaria SState, Berlin, Germany	+12 234 8552 923
3	Audi AG	--	--
3	Honda Motor Corp	--	--

However, clicking on the cars tab, should give more options to the user, including the ability to filter cars by manufacturer, and perform a number of actions such as view details of a car, create a new car record, edit an existing car record, or even delete it using the action buttons located in the last table column:

CAROZZA APP

Manufacturers Cars

All Cars Models

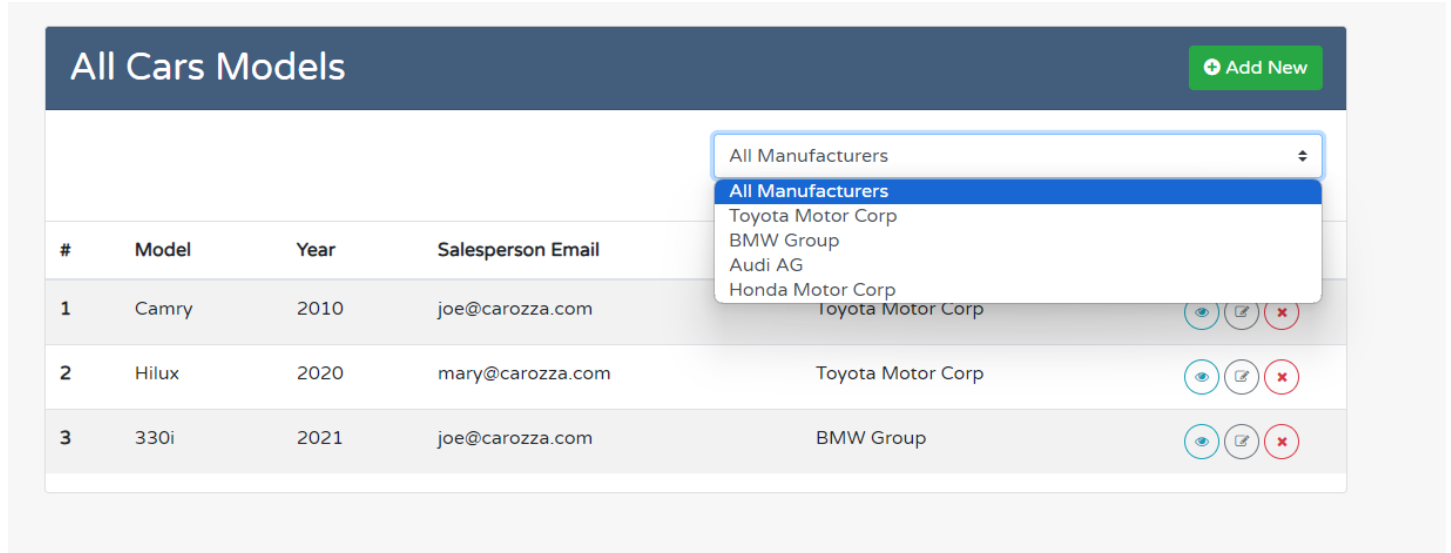
Add New

All Manufacturers

#	Model	Year	Salesperson Email	Manufacturer	Actions
1	Camry	2010	joe@carozza.com	Toyota Motor Corp	<div> <div></div> <div></div> <div></div> </div>
2	Hilux	2020	mary@carozza.com	Toyota Motor Corp	<div> <div></div> <div></div> <div></div> </div>
3	330i	2021	joe@carozza.com	BMW Group	<div> <div></div> <div></div> <div></div> </div>

C2. Filtering list with dropdown menu

The user should be able to filter and display only the cars of a particular manufacturer using the dropdown menu in the main view by visiting <http://<your-ip-address>/cars>:

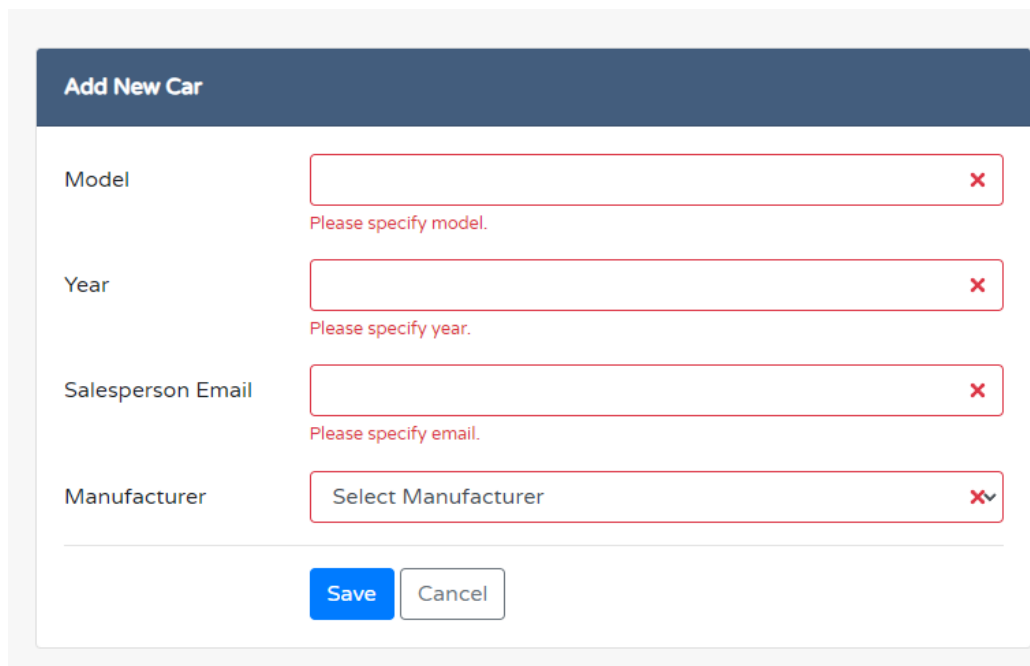


The screenshot shows a web application titled "All Cars Models" with a green "Add New" button. A dropdown menu is open, displaying a list of manufacturers: "All Manufacturers", "Toyota Motor Corp", "BMW Group", "Audi AG", and "Honda Motor Corp". The table below lists three cars:

#	Model	Year	Salesperson Email	Manufacturer	Actions
1	Camry	2010	joe@carozza.com	Toyota Motor Corp	View Edit Delete
2	Hilux	2020	mary@carozza.com	Toyota Motor Corp	View Edit Delete
3	330i	2021	joe@carozza.com	BMW Group	View Edit Delete

C3. Add New Car

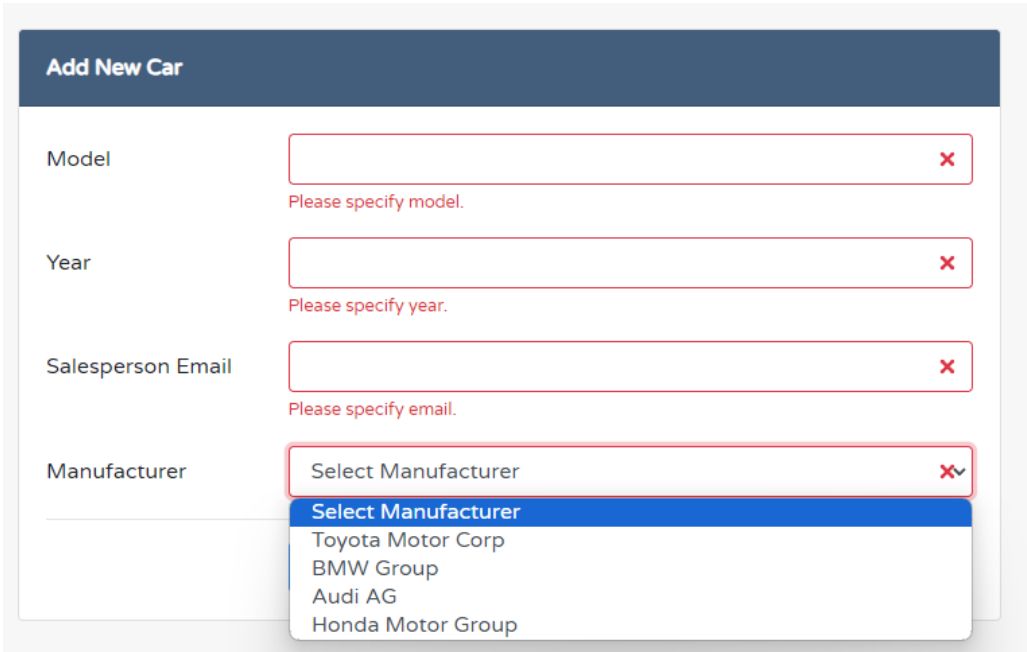
The user should be able to add a new car record using the green Add New button. This button should redirect the user to <http://<your-ip-address>/cars/create>. The form should validate all the inputs by requiring the user to input all the fields, and making sure to add a valid email address, before saving the record to the database.



The screenshot shows the "Add New Car" form with the following fields and validation messages:

- Model:** Validation error: "Please specify model."
- Year:** Validation error: "Please specify year."
- Salesperson Email:** Validation error: "Please specify email."
- Manufacturer:** Dropdown menu with "Select Manufacturer" selected. Validation error: "Please specify manufacturer."

At the bottom of the form are "Save" and "Cancel" buttons.

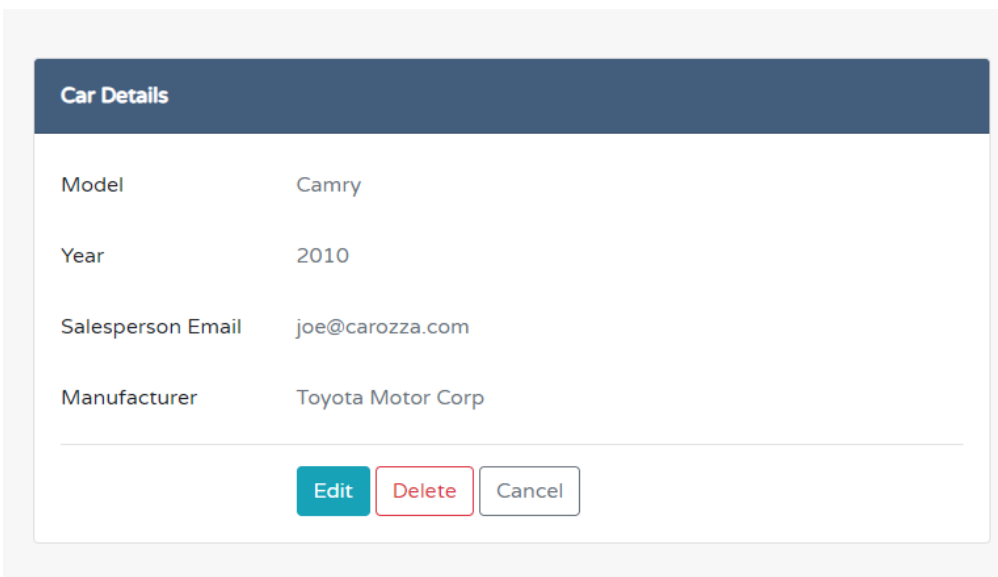


The 'Add New Car' form features a dark blue header. It contains four input fields: 'Model', 'Year', 'Salesperson Email', and 'Manufacturer'. Each field has a red border and a red 'x' icon in the top right corner. Below the 'Model' and 'Year' fields is the text 'Please specify model.' and 'Please specify year.' respectively. Below the 'Salesperson Email' field is the text 'Please specify email.'. The 'Manufacturer' field has a dropdown menu open, showing a list of manufacturers: 'Select Manufacturer' (highlighted in blue), 'Toyota Motor Corp', 'BMW Group', 'Audi AG', and 'Honda Motor Group'. The dropdown menu has a red 'x' icon in the top right corner.

The manufacturer should be selected from a dropdown menu that is dynamically generated from the manufacturers database table in case in the future new deals are made the list of manufacturers is updated.

C4. View Car Details

The user should be able to view all the car details as per the following mockup. Furthermore, this screen should allow the user to edit and delete the record, or else go back to the main homepage using the cancel button. This screen can also be accessed directly through the URL <http://<your-ip-address>/cars/{id}>:



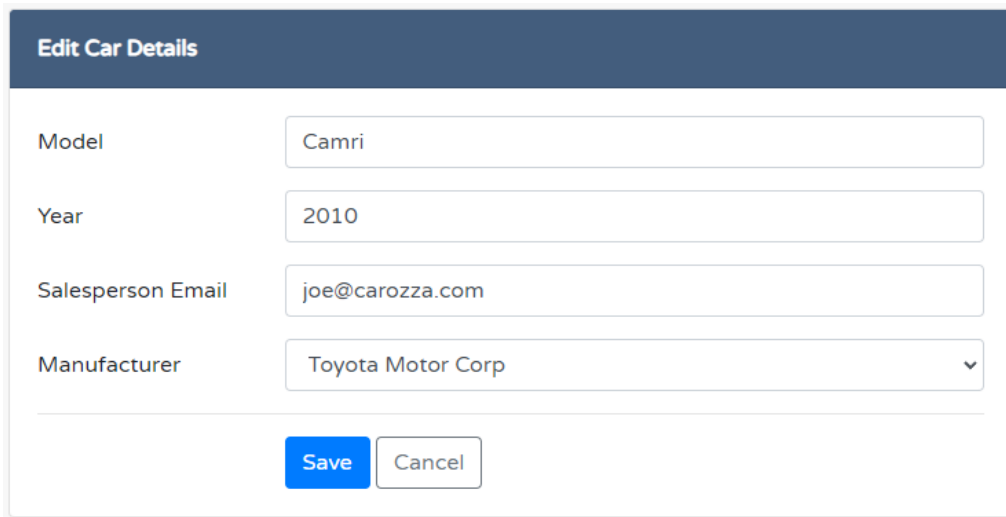
The 'View Car Details' form has a dark blue header. It displays the car details in a table-like format:

Model	Camry
Year	2010
Salesperson Email	joe@carozza.com
Manufacturer	Toyota Motor Corp

Below the table are three buttons: 'Edit' (blue), 'Delete' (red), and 'Cancel' (grey).

C5. Edit Car Details

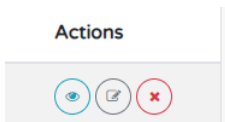
The user should be able to edit the car details using a Laravel form. The form should validate all the inputs as required inputs, including making sure that the email address is valid. This screen can be also accessed directly through the URL <http://<your-ip-address>/cars/{id}/edit>.



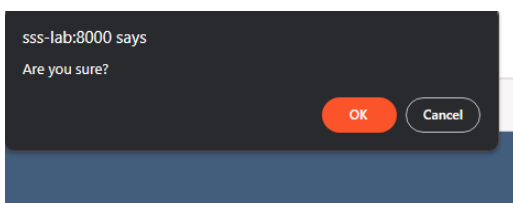
The manufacturer should be selected from a dropdown menu that is dynamically generated from the manufacturers database table in case in the future new deals are made the list of manufacturers is updated.

C6. Delete Car Record

The user should be able to delete a car record either from the show car details screen as per Specification C4, or else by using the delete action button from the home screen:



The user should be prompted for a confirmation before the record is permanently deleted.



Assessment Tasks

For this assessment, you are required to build the Carozza App in Laravel 10.x **as per the specifications described in the previous section**. For each task, remember to create at least one git branch and push it to GitHub.

Task 1 – Database Setup (7 marks)

Setup all the database objects and table associations using database migration files. Create the ORM models and make sure all fields are setup as fillable. (AA4.2, **7 marks**)

Task 2 – Template Layout (9 marks)

Convert the mockup screens in the specification using the Laravel Blade Templating engine. You can use the Bootstrap theme that we used during the Server Side Scripting coursework.

- Routers and controllers setup correctly (SE2.4, **2 marks**)
- Appropriate layout file (AA1.3, **2 marks**)
- Use of includes and yields to extend layout to mockup views (AA1.3, **5 marks**)

Task 3 – Read Data (18 marks)

Setup the necessary views, routers and controllers to dynamically display the manufacturers and car data from the database.

- Routers and controllers setup correctly (SE2.4, **2 marks**)
- Display all cars from database. (KU2.2, **2.5 marks**)
- Display all manufacturers from database (KU2.2, **2.5 marks**)
- Use appropriate named routes to access both screens. (SE3.3, **6 marks**)
- Filtering cars by manufacturer. (KU1.4, **5 marks**)

Task 4 – Create Data (11.5 marks)

Setup the necessary views, routers and controllers to allow the user to create a new car and save it to the database.

- Routers and controllers setup correctly (SE2.4, **2 marks**)
- Use an appropriate named route to access this screen. (SE3.3, **2 marks**)

- Form validation implemented. (KU3.2, **2.5 marks**) (SE4.3, **5 marks**)

Task 5 – Edit Data (13.5 marks)

Setup the necessary views, routers and controllers to allow the user to edit an existing car record and save it to the database.

- Routers and controllers setup correctly (SE2.4, **2 marks**)
- Use an appropriate named route to access this screen. (SE3.3, **2 marks**)
- Correctly loading the old values in the edit form (AA2.3, **2 marks**)
- Form validation implemented. (KU3.2, **2.5 marks**) (SE4.3, **5 marks**)

Task 6 – Delete Data (7 marks)

Setup the necessary views, routers and controllers to allow the user to delete an existing car record and remove it from the database.

- Routers and controllers setup correctly (SE2.4, **2 marks**)
- Using a hidden form to implement the deletion (AA2.3, **5 marks**)