The power of the Web is in its universality.
Access by everyone regardless of disability is an essential aspect.

Tim Berners-Lee, W3C Director and inventor of the World Wide Web

INFO263 Course Project

Redesigning a Web Site with Web Accessibility in Mind

Course contribution: 40%
Due date: 18th October 9 p.m.
Late submission: 20th October 9 p.m. (15% penalty)

Project Objectives

Following are the objectives for this project:

- Learn about building a basic web application, and specifically working with and enhancing someone else's code, using PHP and JavaScript (JQuery),
- Learn about mobile-first web computing aspects, such as responsive design,
- Learn about CSS styling (possibly using Bootstrap or PureCSS libraries),
- Learn about web accessibility considerations and web accessibility evaluation process,
- Improve your collaboration skills with a group.

Project Introduction

The Web Accessibility Initiative[1] states:

> *The Web is fundamentally designed to work for all people, whatever their hardware, software, language, location, or ability. When the Web meets this goal, it is accessible to people with a diverse range of hearing, movement, sight, and cognitive ability.*
>
> *Thus the impact of disability is radically changed on the Web because the Web removes barriers to communication and interaction that many people face in the physical world. However, when websites, applications, technologies, or tools are badly designed, they can create barriers that exclude people from using the Web.*

[1] Introduction to Web Accessibility -- https://www.w3.org/WAI/fundamentals/accessibility-intro/.

Accessibility is essential for developers and organizations that want to create high-quality websites and web tools, and not exclude people from using their products and services.

Imagine that TyreTown is (a fictitious) local business which hired you to improve their business operation according to the modern digital era requirements. You have been tasked with transforming their basic dynamic web page so it complies with the basic web accessibility guidelines. While the TyreTown manager is aware of the drive to improve web accessibility, she does not know how it is to be done, on top of being too busy to think about it. For this project you are required to redesign a dynamic web site with web accessibility guidelines in mind.

Please not the following points:

- You will be starting with an implementation of a website produced by an armature web developer. This implementation is a good stab in the right direction, but web accessibility considerations were left out of the implementation.
- The code you will be given is, to put it simply, dog's breakfast.
  - It is poorly organised: some of the code appears in the wrong place.
  - There is a lot of *dead* code, which is the code not running but simply hanging around waiting to be removed.
  - There is a poor separation between HTML and PHP code, which further limits the improvement of this implementation.
- Thus you are required to understand the code, and refactor the web site with good coding practices in mind.
- You have full freedom (actually, you are encouraged) to rewrite all of the code you are starting with to produce a well-designed implementation.

The project implements a small part of a system intended for moving a business to a paperless business process. The part you will be working with is intended for digitisation of receipts from wheel alignment reports and invoices that come with the reports. The web site includes a dynamic web page, connected to a web server and a database. The code extracts information about a particular invoice from the database, and displays the information to the user. Fortunately, the website has been designed to work on a variety of screens from mobile devices to desktop computers. Although, please keep in mind that this implementation is more of a very rough prototype instead of a fully-functioning web application.

Project Tasks

In this project, you are required to produce the code according to the following requirements:

1. The web page must display invoice information and the inspection details related to a specific invoice. The invoice and inspection information should be retrieved from the database. Please be critical about what page elements need to be coded in static HTML and which elements need to come from the database. As a starting point, consider that PHP code should not be writing any repetitive HTML code, such as **`<tr><td>…</td></tr>`** tags. The code should be organised appropriately to separate

PHP, JavaScript, CSS, and HTML code into separate files, working together as one application. Instead of including the code for the JavaScript and CSS libraries into your codebase, use the content distribution networks to load the necessary JavaScript and CSS libraries on-the-fly. To get an excellent grade, consider the following two features:
    a. Add dynamic styling to the inspection information which will change the text colour of the values based on whether they are within the target range or not. For example, if a value is outside of the target range, it would be shown in red, or, if the value is within the target range, it would be show in green.
    b. Hiding and showing individual columns and/or rows. This can be done by clicking on a column or row header to collapse it; clicking on the header again should reveal the corresponding row or column.
2. Consider a mobile-ready web page: the information should be shown clearly on a mobile device, where the content should not be cut off and thus inaccessible. It should be of appropriate size when viewed on a small screen and without the need to be zoom in on the content. A good example of this is the Mighty Ape website, https://www.mightyape.co.nz/. As you can see, the page elements are adjusted to fit the smaller screen size and are not cut off. A bad example of mobile design is from a Mile High Comics website, http://www.milehighcomics.com/. Depending on what browser you use, you will see different results. You can overcome these issues, by using appropriate libraries (such as Bootstrap) as these libraries were built with mobile browsing in mind. Thus you will not need create CSS styling from scratch. This is quite a common approach to building modern web applications because a huge proportion of website traffic is coming from mobile devices.
3. Explore the online material on web accessibility, then conduct an evaluation of web accessibility of your implementation, and consider the necessary improvements as applicable in the given context. You may want to start by browsing these pages:
    a. Introduction to Web Accessibility – https://www.w3.org/WAI/fundamentals/accessibility-intro/
    b. Evaluating Web Accessibility Overview – https://www.w3.org/WAI/test-evaluate/
    c. A First Review of Web Accessibility – https://www.w3.org/WAI/test-evaluate/preliminary/
4. Write a report documenting your refactoring work and the code, and the web accessibility features of the implementation. Please note that not all web accessibility features are appropriate in this case: choose the features that you think are essential for this project.

Resources

The resources for the project are to be found on UC Learn under the Course Assessment section. Apart from this project specification, the resources include the following:

1. A scanned PDF file `Wheel Alignment Receipt.pdf` of an actual wheel alignment receipt and report. Use this as a baseline to understand what is displayed in the prototype web page implementation. Please note that *it is best not to copy the layout* as

it is shown in the PDF file. Change it as you see fit to the task, but remember to include all of the information, with appropriate interactive features for hiding and revealing the information.

2. An **`assignmentDB.sql`** database dump with **`branch, owner, vehicle, inspection, invoice`** and **`targetvalue`** tables. The **`assignmentDB`** database will not require you to make any **UPDATE** or **INSERT** queries, you should only be using **SELECT** queries for this assignment. The tables have the following meaning:
    - The **`branch`** table contains the information about each branch.
    - The **`owner`** table contains the information about the owners of the inspected vehicles.
    - The **`vehicle`** table contains information about the vehicles.
    - The **`inspection`** table contains the information from wheel alignment inspections and services.
    - The **`invoice`** table contains information about the invoice that the customer would have received, such as the price and quantity.
    - The **`targetvalue`** is a reference table which is used to keep track of what the target values for the inspection should be.
3. An ER Diagram showing the relationships between the tables. The diagram is shown in Fig. 1. Please note that some of the table inspection columns in this diagram are omitted for clarity. Please use MySQL Workbench to find the names of all of the columns.

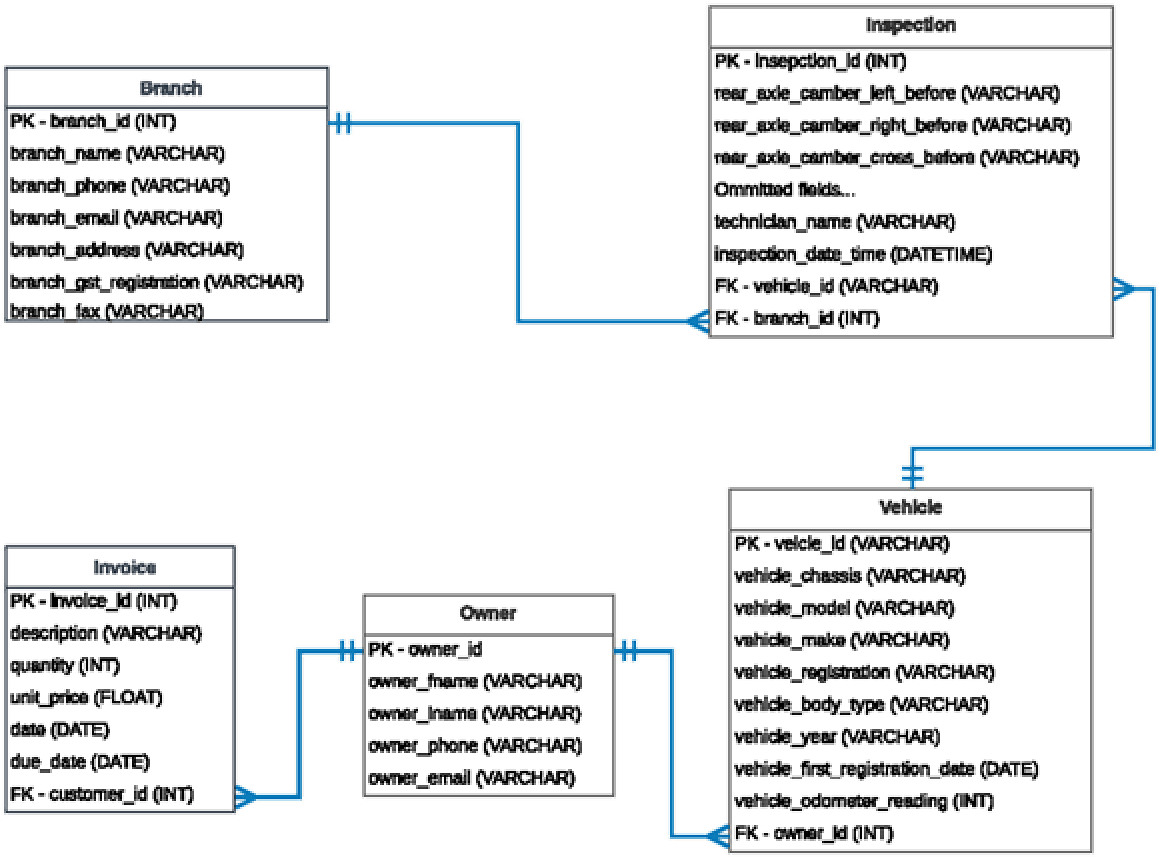

Figure 1. ER diagram showing the structure of the **`assignmentDB`** database.

4. The code of the prototype implementation you will be starting your work with. Bear in mind, this code is missing the Bootstrap library code. Please keep in mind that this code cannot be considered *ready-to-run*. It is your job to understand what to do with, and how to get it going in the first place.

Here are the details for connecting to the database, if you are working on campus:

- Hostname/IP: **`132.181.143.31`**
- Database: **`INFO263_19S2_wheel_alignment`**

These details should allow you to create a new database connection in MySQL Workbench just as it has been done in the tutorials.

Version Control

It is highly recommended that you consider the use of a version control system for collaboration. In any realistic development project it would be unfathomable to rely on manual code backups and email to share the code with your colleagues. If you have not yet come in touch with version control, you may think of this opportunity in terms of getting a skill which bill be invaluable in your future work. Follow this link to find out about version control in general, and, specifically, about the use of Git for version control:

https://git-scm.com/book/en/v2/Getting-Started-About-Version-Control

Alternatively, here's the same information recorded in an online video:

https://git-scm.com/video/what-is-version-control

Here are some practical tutorials on using Git and GitHub follow these tutorials, while presenting the same skills and information in a different way:

- https://rogerdudler.github.io/git-guide/
- https://swcarpentry.github.io/git-novice/

Please note, that it is your individual responsibility to make sure none of your work is lost.

And yet, despite all your best effort, you will most likely find yourself stumped by your code. The most important thing in such a situation is to try and get to this point early, rather than leaving it to the last minute. The second most important thing here is to remember that you are not alone:

https://www.codementor.io/matstc/avoid-frustration-as-programmers-ge54ddszr

If you do choose to use GitHub, make sure your repository is configured as a private repository, so that it is only visible to your group members.

Getting Help

- If you are mercilessly stuck, or if you don't understand some details of the requirements, and specifications, your first point of contact should be your group members. It is expected that you learn from each other and work together quite closely.
- The next thing you might want to do is to post your questions on UC Learn via the INFO263 General Forum. You are encouraged to use the rest of the class as your sounding board, but, needless to say, solutions cannot be posted on UC Learn. You are advised to keep an eye on the INFO263 General Forum messages, as there are likely to be updates and clarifications posted there in due course of time.
- Please feel free to use the standard lab streams to ask Doug for help and advice on the development of your project.
- However, if you have problems within your group, please notify your course coordinator, Constantine, immediately. It is inevitable that some groups will struggle to communicate and collaborate. These issues should be taken into account, noted on record, and, hopefully, resolved as soon as possible.

Submission Expectations

Your group is required to submit the solution code and report. The specifications for the code and report are provided next. All files, including the report, should be submitted as a single zip file.

Code

- Include all the code required to run your website. Since you are not required to alter the data in the database, you do not need to include it. It is essential that your code is organised appropriately, and located in appropriate files, according to best practices: separate static HTML files (if necessary), PHP file, JavaScript files, and CSS files. Most importantly, recall the discussion from lectures on the best practices for mixing HTML and PHP code in .php files. Please avoid using that you avoid using PHP's echo function to send HTML code with spliced variables to the output stream. Please avoid using PHP's heredoc operator. Instead embed PHP code using the short PHP tag.

Report

- The report should contain a clear concise (2-3 pages) description of how you improved the organisation of the code, its presentation, organisation, explicitly separating the client-side and server-side functionality and your decision-making process related to this.
- The report should contain the results of your initial web accessibility evaluation, and the subsequent work you've completed while attending to the identified web accessibility issuer/problems in the initial code (2-3 pages).

- The report should also provide a reflective summary of your group’s teamwork (1 page). For example, please state how you communicated with each other, how you managed and shared your code, how you separated the tasks between team members, what went well in this project and what didn’t go well, and what you would do differently next time around.
- The report should be six to seven pages long, submitted in PDF format, in 10 or 11 pt. serif-family font, with medium margins.

Marking Schedule

Component	**Description**	**Value**
Functionality	Show information for an invoice and wheel alignment. The page must display data from the database.	10
Presentation	Information and the web page should be presented in a way convenient for viewing both on mobile and desktop devices.	10
Dynamic styling	Change the text colour of the values based on whether they are within the target range or not.	5
Interactivity	Hiding and showing individual columns and/or rows.	5
Web accessibility	Selection and implementation of appropriate web accessibility features	20
Code style	Code organisation, separation of HTML, PHP, CSS, JavaScript, formatting, basic commenting, sound code structure, separation into functions and/or classes.	20
Report	Code refactoring and improvements description, web accessibility evaluation and implemented features description, reflective summary.	30
	Total	100