

You, as a developer, have been contacted by WeDeliverTECH™ company. The company specializes in selling technology components and delivering them directly to customers. They would like a web app that has a Reception Management dashboard to manage the following:

- Staff member out-of-office logging
- Deliveries tracking

In a meeting with the company's executives, they presented their needs as follows.

Requirement 1

They require the ability to keep track of staff members who have clocked out of the office. If a staff member should leave the office for a meeting during the business day, the receptionist must “clock them out” on a dashboard. This dashboard will give the receptionist an overview of who is currently in the office and who is currently out of the office.

If a staff member must leave the office (for example, during a lunch break), the receptionist must “clock them out”. She must select the staff member in the table and click the ‘Out’ button. She must then have a way to enter the length of the staff member’s absence in minutes, and the system should calculate and show what the duration of their absence (In hours and minutes if equal or greater than 60minutes) will be in the table, as well as their expected time of return.

This is how the receptionist knows when they should return to the office and that they are out of the office. Their status should also change from In to Out.

Likewise, when they return to the office and are “clocked in” their status must change from Out to In. If the staff member has been out-of-office longer than indicated, a toast notification should appear on the screen with the staff picture, name, surname, and amount of time they have been out of the office. The receptionist can then decide what to do next, clear the notification or inform her manager.

The toast notification that must pop up if a staff member has not returned in time must stay on the screen until the receptionist closes the notification. The notification for a staff member that has not returned in time must appear only once and be cleared or closed by the receptionist.

When the staff member returns from their absence, clicking the ‘In’ button will clear their Out Time, Duration, and Expected Return Time cells and update their status.

Requirement 2

On the same web page, they require functionality to keep track of current deliveries of orders to customers. To provide the customers with the best service, the receptionist must have information regarding the current deliveries. Therefore, there must be a Schedule Delivery area in the dashboard, where the receptionist manually inputs information about the delivery driver and their current delivery when they arrive for work in the morning.

This is necessary as delivery drivers are employed temporarily, and some might not show up for work every day. The driver’s type of vehicle (Motorcycle or Car), name, surname, telephone number, delivery address and the time that each delivery driver will return to the office needs to be captured to ensure the receptionist knows who is not available for deliveries at any point in the business day.

Once the required information is captured, clicking the 'Add' button will add the delivery information to the Delivery Board table. The receptionist has asked that vehicle icons be used in the Delivery Board, as it is easier for her to find the correct delivery at a glance.

WeDeliverTECH™ also requested that if the return time has passed and the delivery driver has not yet returned, a toast notification with the name, surname, telephone, estimated return time, and address of the current delivery must be displayed so that the receptionist can follow up with the delivery driver. When a delivery driver returns to the office, the receptionist can select their row in the delivery board and click the 'Clear' button. This should remove their row from the Delivery Board and delete the object for that delivery driver. There must be a confirmation popup so the receptionist can't accidentally remove a delivery driver.

Requirement 3

The Current Date and Time should be updated every second at the bottom of the web page. This is essentially a digital clock for the Receptionist to refer to. The specific format required for this clock is "Day Month Year Hour:Minute: Second".

Requirement 4

The executives have indicated that the company brand must be reflected in this system. This should be done by including the WeDeliverTech™ logo on the web page and only using the specified fonts and colors included in the Company Branding Profile for the web elements.

Instructions

Mock-up

WeDeliverTECH™ has provided us with a basic mock-up of the layout they would like the reception management dashboard to follow:



Reception Management Dashboard
Staff

| Picture | Name | Surname | Email address | Status | Out Time | Duration | Expected Return Time |
|---------|--------|-----------|------------------------|--------|----------|-----------|----------------------|
| | Dennis | Nicholson | dennisN@example.com | Out | 12:00 | 1hr 20min | 13:20 |
| | Ivan | Garza | ivan.garza@example.com | In | | | |
| | | | | | | | |

Out

In

Schedule Delivery

| Vehicle: | Name: | Surname: | Telephone: | Address: | Return time: |
|----------------------|----------------------|----------------------|----------------------|----------------------|----------------------|
| <input type="text"/> | <input type="text"/> | <input type="text"/> | <input type="text"/> | <input type="text"/> | <input type="text"/> |

Add

Delivery Board

| Vehicle | Name | Surname | Telephone | Delivery Address | Return time |
|---------|-------|-----------|-----------|------------------|-------------|
| | Frank | Bester | 123456789 | 3 Nowhere Lane | 13:00 |
| | Anne | Katherine | 123456781 | 4 Somewhere Road | 11:00 |

Clear

(current date and time)

Dashboard Layout

Your task as the developer is to ensure that the correct functionality is present according to the customer requirements stipulated by the executives of WeDeliverTECH™.

Instructions

Project specifications

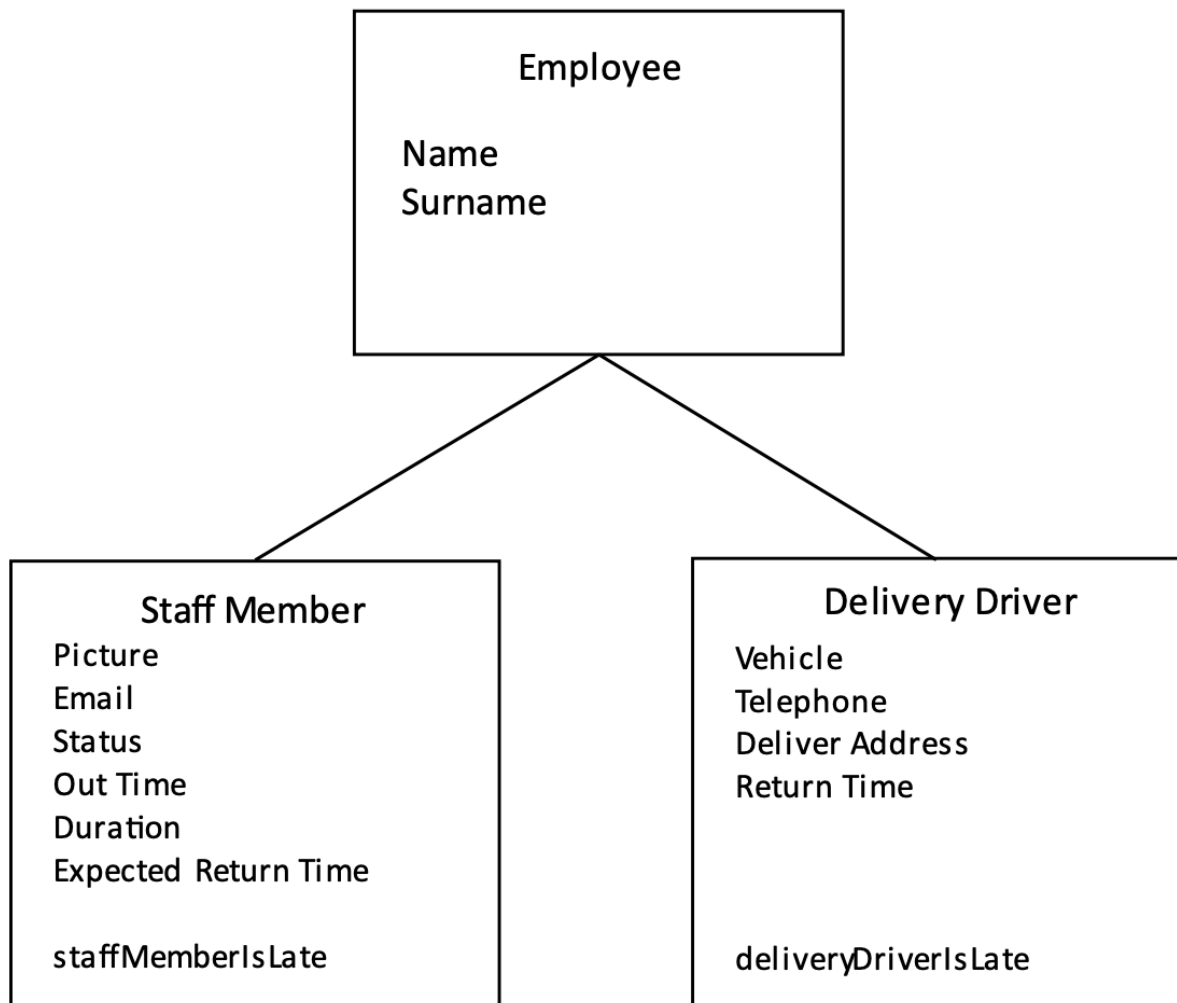
The WeDeliverTECH™ executives have also indicated that their internal IT department will maintain the web application once it has been delivered. The head of the IT department requested that Bootstrap with JavaScript/jQuery and OOP (Object Oriented Programming) concepts should be used for the web application.

As the IT department does not know NodeJS, they have requested that NodeJS and NPM should not be implemented for the project. Any external libraries or plugins may be used for the project, but the minimum requirements (Bootstrap, JavaScript/jQuery, and OOP concepts) must be met.

Remember to include instructions in your read.me file for using external libraries or plugins. All JavaScript functionality described in this web application's requirements must be in a JS file in the correct folder. This file needs to be named wdt_app.js. You should pay attention to the user experience. Therefore, when buttons on the Dashboard are hovered over with a mouse cursor, this should be indicated with some animation on that button.

You may add animations to the web application where you think it may enhance user experience. To protect their staff members' identities, the company has requested that you use the API (Application Programming Interface) <http://backend.restapi.co.za/items/staff> to get the demo information for five (5) staff members.

The IT Department has provided this diagram depicting the required types of objects with methods or properties:



The Staff table should then be filled with the newly created objects (created according to the diagram) when the web page is initially loaded.

The five (5) staff members in the table should be unique (i.e., five (5) different staff members).

The receptionist will manually fill in the Delivery Driver information, and all user input data in this system should be validated to ensure it is in the correct format. The minimum validation should be that all fields are filled in with the correct data type and the correct time format (hh:mm).

This input data should be used to create the relevant Object, and the Delivery Driver table should then be filled with this object's data.

There has also been a request to create and maintain a project plan using the Jira software. Sprints, Epics, and Issues must be present, along with a Board that is kept up to date regarding task progress and task assignments (which team member is assigned to each task). The timeline for this project should match how long students have been given to complete this Semester Project (i.e., total project time is four (4) weeks for full-time students and eight (8) weeks for part-time students).

Although this is just the first phase, the executives plan to include more functionality in this project's coming phases. Because of this, they have requested that the web application has a navigation bar at the top of the screen with the following menu options:

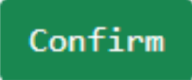
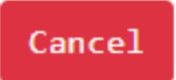
| Dashboard | Inventory | Orders |
|-----------|-----------|--------|
| | Search | Search |
| | Add | Add |
| | Remove | Remove |

The Inventory and Orders sub-menu items, which should be hidden unless the navbar is interacted with, should not be operational, meaning nothing should happen when you click on them. The 'Dashboard' navigation item should be active by default. Clicking on this navigation item will navigate to the main Reception Management Dashboard page with the tables for staff and delivery.

WeDeliverTechTM Company Branding Profile

Logo: *(The logo will be in the repository)*

| Element | Style |
|--|---|
| Web page text elements (paragraphs, etc) | Font: Calibri Font colour: #212529 |
| Navbar | Font: Consolas |
| Navbar Menu Item | Background colour: #0E8EA8 Font colour: #fff |
| Navbar Sub-menu Item | Background colour: #83D1E1 Font colour: #212529 |
| Table | Border colour: #212529 Border corners: rounded Border thickness: thin lines |
| Table Header | Font: Consolas Background colour: #0E8EA8 Font colour: #fff |
| Table Cell | Font: Calibri Background colour: #83D1E1 Font colour: #212529 |
| Positive button | Font: Consolas Background colour: #198754 |

| | |
|--|---|
|  E.g., Confirm, Next, Add, Yes | Font colour: #fff |
| Negative button  E.g., Cancel, Previous, Delete, Remove | Font: Consolas Background colour: #dc3545 Font colour: #fff |




Instructions

Jira project management

You must create a “Reflection Report” in the form of a PDF document named “Jira.pdf” This report must include the following:

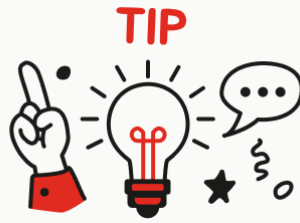
- Screenshots of the project Sprints, Epics, and Issues, elaborating on why you chose to create these issues.
- Screenshots of the project Timeline, Board and Backlog, and elaborations on the progress of the project
- A summary of the management of this project, explaining why some choices were made in developing this web application. This section can discuss some of the more difficult tasks and how the results were achieved. The reflection report section must be between 500-1000 words and can include additional screenshots of what is being addressed.

Example of private repository folders

| | |
|---|---------------------|
|  Documentation | Create Jira.pdf |
|  Web Application | Updated Staff Table |
|  README.md | Initial commit |

Instructions

Important considerations



Take note of the following points when working on the semester project.

System styling

System styling must adhere to WeDeliverTech™ Company Branding Profile.

All the styling should be in a single, separate CSS file.

Correct API call

Correct API call/s made on page load.

The API JSON String response should be converted into a JS object/s, then used to create relevant class objects.

Inheritance is used in object creation, using the data from the API call.

Staff table populated with the objects of the five (5) unique staff members.

There must be a `staffUserGet` function that makes the API call(s) and processes the response(s), i.e., converts the API response(s) to the relevant JS class object(s).

Click out prompts

Clicking 'Out' prompts the user for data, updates the relevant staff member's object, and then updates the Staff table from the object.

There must be a `staffOut` function.

Click in prompts

Clicking 'In' updates the relevant staff member's object and updates the Staff table from the object.

There must be a `staffIn` function.

Handling late staff members

Toast should be shown, with the correct information, when a staff member has not returned by the expected return time. The notification should appear only once, and the receptionist must close or clear the notification.

There must be a `staffMembersLate` function.

Delivery Driver table

Delivery Driver information is manually entered into input elements in the Delivery Driver table. The table is populated with the Delivery Driver object data.

There must be an addDelivery function that adds the delivery driver's information to the Delivery Board table.

Validate Delivery Driver input

Delivery Driver input is validated (Checked for correct format).

There must be a validateDelivery function.

Inheritance

Inheritance is used in the Delivery Driver object creation. Appropriate icons are used for Vehicle types, not images.

Handling late delivery drivers

Toast should be shown, with the correct information, when a delivery driver has not returned by the estimated return time.

There must be a deliveryDriverIsLate function.

Date and Time

The current Date and Time should be updated every second (basically a digital clock) in the specified format (Day, Month, Year, Hour:Minute: Second". E.g. 5 June 2022 14:54:22 or 05-06-2022 14:54:22

There must be a digitalClock function.

Hover animations

At least the minimum hover animations are present, as requested.

Submission

Submission Requirements

Code

- Your **application code** must be pushed to your GitHub Classroom repository **BEFORE** the Semester Project deadline.
- After the deadline, you cannot commit any more code to your repository or Git classroom.
- Any attempts to commit after the deadline will be automatically rejected.

Moodle

- Create a .txt file on your computer, which must contain **your GitHub username** and the **URL link to your GitHub repository**.
- Name the .txt file as follows: "**FName_LName_SP1_CA_ClassXXYY.txt**"
 - Replace 'Class' with your class, e.g. 'Aug', 'Oct', etc.
 - Replace 'XX' with your class year e.g. 22, 23.
 - Replace 'YY' with either FT for Fulltime, or PT for Parttime.
 - **JOHN_DOE_SP1_CA_JAN23FT.txt**
- This .txt file **must be submitted** on the [Moodle Semester Project 1 Submission](#) page.

Example contents of a submission txt file

GitHub Username: JohnSmith

Repository: <https://github.com/johnsmith/jan23ft-sp1-ca-johnsmith>

Grading

The Semester project will be **graded "A to F"**.

- F - 0-39 - Failed the course and have to do a Resit.
- E - 40-49
- D - 50-59
- C - 60-79
- B - 80-89
- A - 90-100



Important reminders

- Only commits made to the Main branch of your created private Git Classroom repository will be graded.
- Any repositories you create outside the provided Git Classroom link will not be graded.
- Ensure that at least one commit is made to your private Git Classroom repository before the weekend of the submission deadline.
- Commits or submissions past the deadline will not be considered for grading.
- Late submissions will not be accepted, and there will be no exceptions to this rule.
- Failure to submit the .txt file to Moodle OR push (commit) your code to the Git classroom will result in a not passed grade.

