INTRODUCTION

The interface is the medium through which the user communicates with the web. This makes the ability to easily support interface updates and feedback critical to the success of a website. The jQuery library concentrates exclusivly on reading and from and writing to the DOM which underlies the HTML and CSS interface of a website.

PROJECT OUTLINE

Create a website for an online booking system for airline flights using jQuery. Version control this project with Git.

Background and Context:

The website you have been tasked with creating will allow users to see flight details and book flights through a form. You will include this functionlity by adding jQuery to an HTML and CSS site.

What you will do:

You have been instructed to create the website for the flight booking system. You will receive an HTML and CSS project for the airport, which includes the following pages:

- A Home page
- A Flights page
- A Purchase page

Each of these pages inludes a link to a blank JavaScript file (`home.js`, `flights.js`, or `purchase.js` depending on the page) as well as a link to `main.js` which contains code that is active on all the pages. You must add functionality to the site by updating only these JS files (do not change the HTML or CSS files you have been provided)

All the pages should include the following functionality:

- When the cursor hovers over the logo in the top left corner of the screen, the image of the plane should rotate by using the `plane-rotated.svg` instead of the `plane.svg` image.

The Home page should include the following functionality:

- Once the page has finished loading (the document is ready) the text of the `h1` element on the page should change from "Airport" to "Welcome to the Airport!"
- The `.groups div` should become an accordion. Each `.group div` is a section in the accordion. The accordion should be completely collapsable.

The Flights page should include the following functionality:

- Clicking a `.flight div` should toggle the selected class on the flight, changing the heading text from yellow to blue and back again.
- Filtering by "All flights" should show all flights.
- Filtering by "Domestic flights only" should hide all international flights and show all domestic flights.
- Filtering by "International flights only" should hide all domestic flights and show all international flights.
- Filtering by "SAA flights only" should show all flights with an `data-airline` attribute of `SAA` and hide all others.
- Filtering by "Flights to and from Durban only" should show all flights with either a `data-origin` or `data-destination` attribute of DUR and hide all others.
- Filtering by "Afternoon flights only" should show all flights with a departure time between 12:00 and 23:59 and hide all others.

The Purchase page should include the following functionality:

- Clicking the PURCHASE button should add a new entry to the table, displaying the flight code of the selected flight, the number of tickets entered, and the total price.
- The total price is equal to the `data-price` attribute of the selected flight multiplied by the number of tickets entered.
- Clicking the REMOVE button should delete all the rows from the body of the table.

The project must be a valid Git repository, with at least 3 commits and a `.gitignore` file. During your demonstration you will be asked to:

- Prove that the project is a Git repository with at least 3 commits
- Switch to an older commit
- Switch back to the newest commit
- Make a change and add it to the staging area

- Commit the new change

Make sure you know the Git commands to do this.

What you will be given:

You will receive the HTML, CSS, JS, and image files of the airport project. Update only the JS files.

PROCESS

This section of the brief contains helpful guidance and information concerning your process. You will be technically guided through each phase of the project.

Phase 1: Planning

Plan the implementation details for your functionality by doing some jQuery reseach.

Phase 2: Implementation Phase

Initialise the project as a Git repo. Update the JavaScript files, commiting the changes as you go.

Phase 3: Testing phase

Ensure that your implementation performs the functions specified in this brief.

Phase 4: Final Phase

You must submit your final project and write the test.

SUBMISSION

- NO LATE SUBMISSIONS WILL BE ACCEPTED
- Submission will take place in Week 8
- Projects should be digitally submitted on Google Classroom
- All deliverables must be placed in a single zipped archive
- The zipped folder must follow the naming convention **`jQuery_xxxxxx_yyyyyy.zip`** where **xxxxxx** is your student number and **yyyyyy** is your first name.

OUTCOME

By completing this project, learners will have learned to:

- Version control a project with Git
- Read from DOM elements with jQuery
- Write to DOM elements with jQuery
- Hide and show DOM elements according to conditions with jQuery
- Add and Remove new DOM elements with jQuery
- Manage their time

ASSESSMENT

Learners will be assessed on their:

- Technical skill
- Technical understanding

- Correctness in implementation
- Professionalism

Please see the attached marksheet for a detailed breakdown of the assessment weightings.

SCHEDULE

WEEK	DATE	CLASS WORK	MILESTONES
1	2017-04-17 to 2017-04-21	• Introduction to jQuery • Introduction to Git	• Planning Phase
2	2017-04-24 to 2017-04-28	• Contact Week	• Contact Week
3	2017-05-01 to 2017-05-05	• Manipulating the DOM • Git config and init	• Implementation Phase
4	2017-05-08 to 2017-05-12	• Show/Hide and Add/Remove Class • Git ignore and status	• Implementation Phase
5	2017-05-15 to 2017-05-19	• CSS, hover, and animate • Git add and remove	• Implementation Phase
6	2017-05-22 to 2017-05-26	• jQuery-UI • Git commit and log	• Implementation Phase • **Progress milestone 1**
7	2017-05-29 to 2017-06-02	• Tips and Tricks • Git checkout	• Testing Phase • **Progress milestone 2**
8	2017-06-05 to 2017-06-09	• Submission, demonstration, and assessment • Test	• Final Phase