

A Simple Flight Search Website

Overview

This task is intended to evaluate your basic web development skills, alongside your PHP and MySQL knowledge. Please read it over carefully before you begin!

You are required to create a simple one-way flight search website. The website will consist of three web pages in the customer's journey through the website:

- A page asking them for their search requirements.
- A script that searches for the customer's flights.
- A flight result display for the flights available.

You are required to create these pages from scratch, which will involve the following technologies and techniques:

- PHP coding
- HTML
- CSS styling
- Calling an API and parsing XML
- MySQL database interaction
- Javascript and AJAX

At a minimum you'll want to have a PHP + MySQL development environment set up on whatever computer you're doing this exercise on!

Please **do not** use a PHP framework for this. We're interested in seeing what actual PHP knowledge you have. For the Javascript portion feel free to use jQuery.

Some files (images, MySQL dumps) have been included for you for to use with this exercise in the zip file you've been provided with. After you're done with the exercise, please zip all of your code up again and send it back so we can review it.

Each section of this exercise gives you a guide to the amount of time you should need to spend on it. Please don't spend all day working on this! These sections are detailed on the next few pages.

Before you begin, make sure you've retrieved a valid API key from

<https://www.vibe.travel/interview> - if you're reading this document you probably already do, but if it has expired feel free to request a new one.

Create a search form (search.php, 10 minutes)

The search page should be a simple HTML form that captures information from the customer. The three main requirements of the form are:

- Departure airport
- Destination airport
- Date

The departure and destination airports should be submitted as a three letter airport IATA code.

A large list of these IATA codes is available in the `airports` table in the database table provided, but for the purposes of this exercise the following examples can be used:

LHR – London Heathrow, United Kingdom

AMS – Amsterdam, Netherlands

CDG – Paris Charles de Gaulle, France

JFK – New York John F Kennedy, USA

SYD – Sydney, Australia

The date can be submitted in any format you wish, but note that the chosen value will be used in the next part of the exercise.

The form should submit your parameters via POST to your next script (searching.php), which will perform a search on the flight web service.

Important Note: The layout of your search page is **not important**, and you will not be assessed on the styling of your search form. All we are evaluating here is your knowledge of how HTML forms work.

Searching for flights (searching.php, 25 minutes)

Once the customer has entered their requirements, you should use these to request matching flights from the flight web service, located at <https://www.vibe.travel/interview> - see appendix 1 for full details.

The flight web service will return an XML document. You will need to parse this document and extract the flight information. This information should be saved into a database - an example MySQL table schema for flight results has been provided.









Once the flights have been inserted into the database, you should redirect the customer to their search results.

The flight results table has a `searchid` column. You should use this to track the customers search between your search script and the results page. The flight web service returns a unique identifier that can be used for this purpose.

Important Note: This script does not need to output any HTML to the customer, and should only redirect them to your next script (results.php). We want to see that you can save and later retrieve the search results, not just parse them straight away.

Search Results (results.php, 25 minutes)

You should now retrieve the customer's flight results that you just inserted in to the database. You should output these and create an external CSS stylesheet to display them as closely as possible to the following screenshot:

	07:45 20 May London Heathrow	10:45 21 May New York John F Kennedy	 Economy	£357.41 per person
	16:25 20 May London Heathrow	20:00 21 May New York John F Kennedy	 Economy	£358.87 per person
	19:50 20 May London Heathrow	20:00 21 May New York John F Kennedy	 Economy	£365.21 per person
	23:00 20 May London Heathrow	12:55 21 May New York John F Kennedy	 Economy	£374.49 per person
	09:50 20 May London Heathrow	12:50 21 May New York John F Kennedy	 Economy	£390.65 per person

Airline logos have been provided for you in the zip file. You can use these to aid you in your flight results display.

Important Note: Obviously the number of results you have and the flight details will be different! If for any reason you're unable to perform the API/MySQL steps, please try to just recreate the above layout using the data above.

AJAX Search Results (all, 10 minutes)

Many modern sites use AJAX to update page elements without a full page reload. You should now try to emulate this using the work you have already done.

- Update search.php to use AJAX methods to send the search data to searching.php, instead of redirecting away from the page.
- Using a POST or GET parameter, make searching.php return the search ID when the search is completed, instead of redirecting away from the page.
- Use this search ID in search.php's AJAX methods to request data from results.php
- Display the formatted data from results.php in a div on search.php. You may want to use a condition to prevent any unnecessary HTML from outputting. Don't forget to include the stylesheet in search.php!

SQL questions (5 minutes)

Based on the database structure you have been provided for this exercise, please provide SQL queries for the following pieces of data:

1. The price of the shortest flight ever found from the web service.
2. The number of flights that have been found that arrive in the USA.
3. The airport name and country code that the cheapest flight arrives at.

These can be included in any script (although the results page might be the most relevant place for them).

Appendix 1 – Flight Web Service

This web service is very simple. By submitting either a GET or POST request including certain parameters, an XML document is returned with flight information.

The web service responds immediately and all the flight data is fictitious. This is not indicative of real world travel web services, which often take up to half a minute to respond.

By receiving an instantaneous response, it should allow you to implement the web service directly in to your customer page flow and avoid any “please wait...” screens.

All requests should be submitted to <https://www.vibe.travel/interview> with the following GET or POST parameters:

-- apiKey – the API key you retrieved from <https://www.vibe.travel/interview> when you got this document.

-- departureCode – 3 letter airport code the flight departs from.

-- destinationCode – 3 letter airport code the customer wants to travel to.

-- departureDate – A unix timestamp of the date the customer wishes to depart.

In response, you will receive an XML document in the following format:

```
<Flights searchid="(unique identifier referencing this search)">
  <Flight>
    <DepartureCode>(three letter code)</DepartureCode>
    <DestinationCode>(three letter code)</DestinationCode>
    <DepartureDateTime>(unix timestamp the flight departs at in local time)</DepartureDateTime>
    <ArrivalDateTime>(unix timestamp the plane arrives in local time)</ArrivalDateTime>
    <Airline>(a 2 letter code designating the airline for this flight)</Airline>
    <Price>(a floating point to two decimal places cost for the flight)</Price>
  </Flight>
  ...
</Flights>
```

The <Flight> element may be repeated a number of times. Each element will provide details of a different available flight.

If there is an error you will receive the following response:

```
<Error>error text</Error>
```

Each unique set of search parameters will conveniently return the same results. This should enable easier testing of the results display.

Appendix 2 – MySQL Table Structure

``airports`` table

- id – The primary auto increment key.
- code – The airport's official 3 letter IATA airport code.
- name – A string naming the airport
- countrycode -- 2 letter country code

``flights`` table

- id – The primary key, auto incrementing.
- searchid – A unique reference to show which search this flight is part of.
- departureairport – The 3 letter airport code the flight leaves from.
- destinationairport – The 3 letter airport code the flight lands at.
- departedatetime – A unix timestamp for the departure date and time.
- arrivaldatetime – A unix timestamp for the date & time the flight lands.
- airlinecode – The two letter airline code for this flight.
- price – A floating point cost for this flight.