## Test the updated MVC framework

You can now test the updated MVC framework, see Figure 12. Note: Again, the url is: ‘http://localhost:8888/3D\_Apps\_2019/lab7/index.php/’, since I am running on a localhost on my Mac, but it will be different in your case. If you are using ‘public\_html’ folder on your ITS hosting space, your URL should be something like this:

* users.sussex.ac.uk/~your-user-name/3dapp/lab7/index.php

You should take the time to read around PHP, and particularly the MVC design pattern or architecture approach. There are formal MVC frameworks already built that you can exploit for both Java (e.g. Struts) and PHP (e.g. Codeignitor). Do a Google search and explore more. You will be expected to adopt this approach in your Assignment (hand in week 12).

Although we are using images so far, you should be able to see that you could just as easily organise the path to your stored 3D models in the backend so that they are rendered in the front end like the images. You would just need to organise the appropriate HTML for instantiation the X3D code at the front end, see Lab 6.

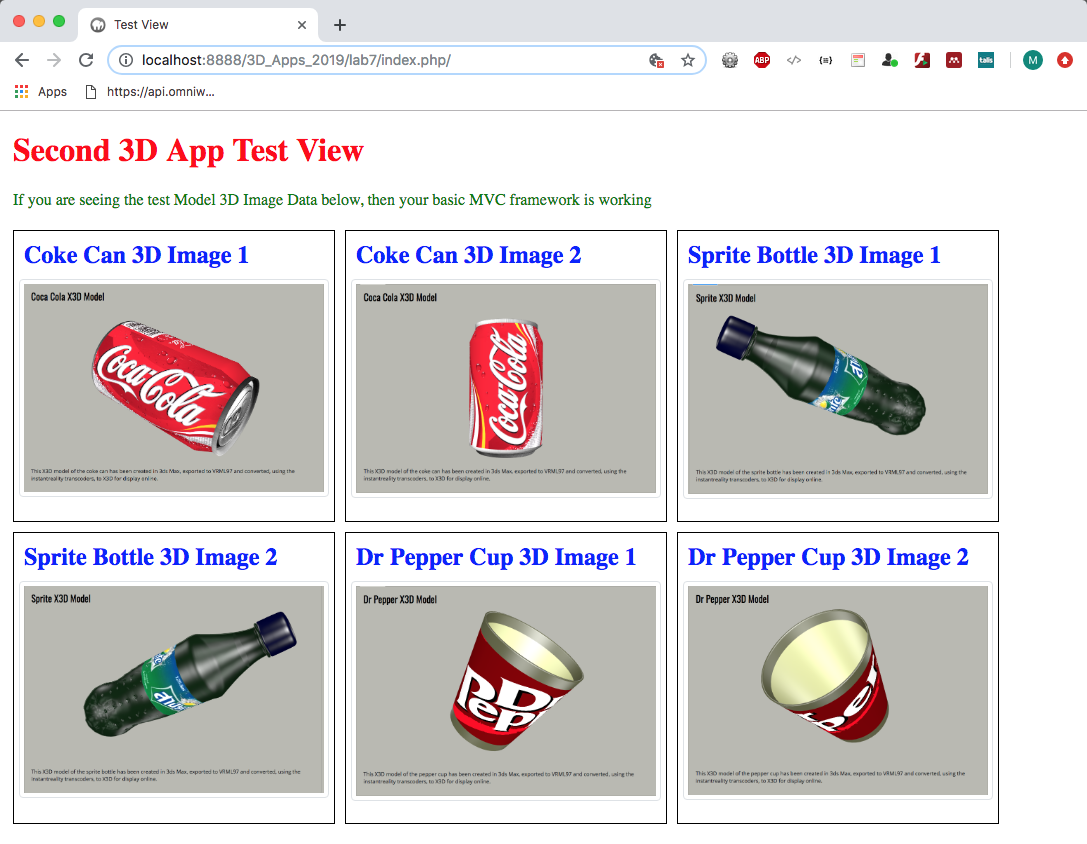


Figure 12: The second 3D App test view

# Part B — Adding SQLIte with PHP to the MVC Framework

This tutorial requires a very basic knowledge of PHP programing and MVC frameworks. If you have completed Part A of this tutorial then go ahead and complete Part B, but if you haven’t finished Part A, then we strongly recommend that you complete this now because Part B will be building on the source code we developed in Part A.

Part B of this tutorial describes how to use SQLite with PHP in your MVC framework. SQLite is a very convenient executable database, more precisely:

“SQLite is an in-process library that implements a [self-contained](https://www.sqlite.org/selfcontained.html), [serverless](https://www.sqlite.org/serverless.html), [zero-configuration](https://www.sqlite.org/zeroconf.html), [transactional](https://www.sqlite.org/transactional.html) SQL database engine. The code for SQLite is in the [public domain](https://www.sqlite.org/copyright.html) and is thus free for use for any purpose, commercial or private. SQLite is the [most widely deployed](https://www.sqlite.org/mostdeployed.html) database in the world with more applications than we can count, including several [high-profile projects”.](https://www.sqlite.org/famous.html)

As mentioned, to become more familiar with MVC concepts we are creating our own MVC framework, and we envisage the need to make it dynamic and capable of storing local data. There are many ways to store data, for example we could use the JSON file approach that we used in Lab 6. There are other solutions, for example we could use a flat file system and store our data in XML files, build an XML parser, and so on the retrieve the data, or we could use AngulaJS to bind data from a file to elements in the web page, we could even use Google’s Firebase.

But, a more flexible way, and one where we can make queries on that data using, for example SQL (Structured Query Language), is to use a database. To meet this need we will connect with the aforementioned SQLite database using PHP. Our PHP code to do this will largely cover several functionalities:

* Connection to the database
* Creating tables in the database
* Inserting data into the database tables
* Retrieving data from the database tables
* Deleting data from the database tables
* Closing the connection to the database

We will also introduce [PHP Data Objects](http://php.net/manual/en/intro.pdo.php) (PDO) as a data-access abstraction, “which means that, regardless of which database you're using, you use the same functions to issue queries and fetch data”[[1]](#footnote-1).

## SQLite

Why use SQLite as opposed to any other database? Putting aside the description of SQLite above as a good enough reason, we could consider using the University web server’s MySQL database, but that is impractical for teaching purposes. I think the lab machines now have a local host installed, so we could use an installation of a localhost with its MySQL (or SQLite) installation. But, it is easier to use your ITS Web Space, which is already set up for PHP and SQLite.

SQLite is a relational database management system developed in the C language. This is an extremely small size database management system — roughly about 350 Kbyte in size. Every time we create a new database using SQLite, it generates a small portable executable file that can be copied over to a different location.

This means we can locate our SQLite database, effectively a C executable, in your public\_html space (or on your local host).

### Building an MVC Framework using PHP and SQLite

1. http://php.net/manual/en/book.pdo.php [↑](#footnote-ref-1)