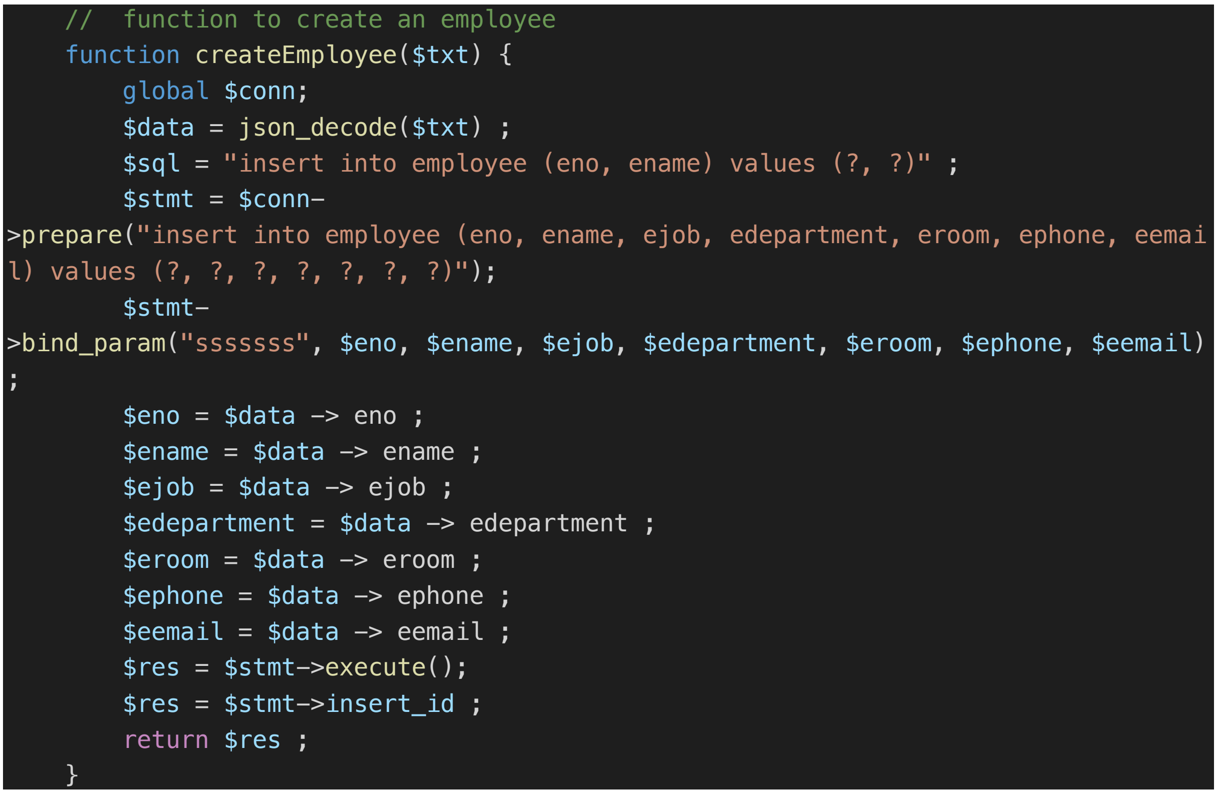
Pick one of the API methods and describe how this works.

The API method that I have chosen to write about for the Critical Appraisal this week is the createEmployee method. This method, along with the form that serves as the front end, allows for a new employee to be created and added to the *employees* database table. The code for this API method is as follows:



```
    //  function to create an employee
    function createEmployee($txt) {
        global $conn;
        $data = json_decode($txt) ;
        $sql = "insert into employee (eno, ename) values (?, ?)" ;
        $stmt = $conn-
>prepare("insert into employee (eno, ename, ejob, edepartment, eroom, ephone, eemai
l) values (?, ?, ?, ?, ?, ?, ?)");
        $stmt-
>bind_param("sssssss", $eno, $ename, $ejob, $edepartment, $eroom, $ephone, $eemail)
;
        $eno = $data -> eno ;
        $ename = $data -> ename ;
        $ejob = $data -> ejob ;
        $edepartment = $data -> edepartment ;
        $eroom = $data -> eroom ;
        $ephone = $data -> ephone ;
        $eemail = $data -> eemail ;
        $res = $stmt->execute();
        $res = $stmt->insert_id ;
        return $res ;
    }
```

First, the user of the website uses the createemployeerec.php page, (which is of the *controller* type) which is the form that has text boxes for all attributes that are going to be inserted into the database table. Once this is run with all the relevant boxes filled in, the above function is called, which first of all calls the database connection script, and creates a new database connection, which will be used to insert the new employee via. The json_decode part converts the request into an object, Afterwards, a new object is created, with the fields eno, ename, ejob, edepartment, eroom, ephone, and eemail., Initially these are just set up to be null. However, after that these database table attributes are then one by one merged with the data that has been entered from the form on the previous page.

Is using an API a good idea? If so, explain why.

Generally, an API (application programming interface) is a set of subroutines, functions, and tools for software and web development. Using an API is indeed a good idea, because in regard to the Tutorial this week, each function can be called if/when they are needed, instead of inserting the same code over and over, for the same web pages.

One advantage of using an API for web development is that it makes updating and amending website code easier, as it can be changed once, in the API file(s), and don’t need to be updated in the *controller* and *view* classes, saving time, as well as being easier overall to make changes. For example, if there was a mistake in the code, it could be changed in the

API file and that would be it fixed, whereas if an API wasn’t utilised then the developer would have to manually trawl through all of the code for the entire site for all instances of the same section of code.

And because the API can be called globally, they can be used throughout the website, and you can use multiple APIs from anywhere online (provided they are of use to the needs of your web application). For example, some common web APIs that are used

In short, APIs can allow for easier access and integration with websites, of which some (websites) provide information and help on using their APIs for their sites, with the aim of allowing for easier access to their data and services. Some sites of these include Google, Twitter, and the online music database website Last.fm.