Advanced Higher Computing Science

Web Development

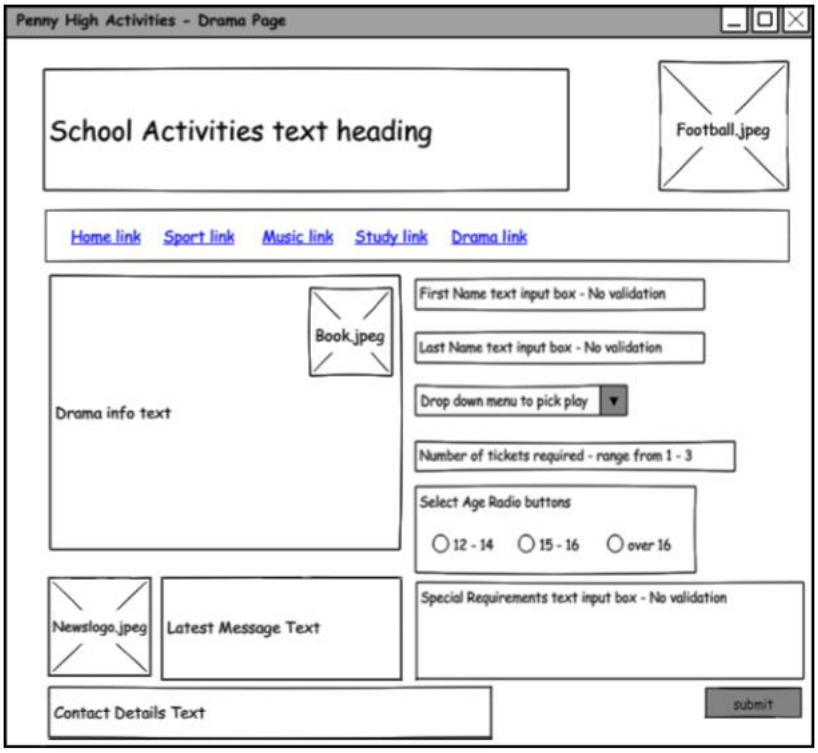
Forms & Validation

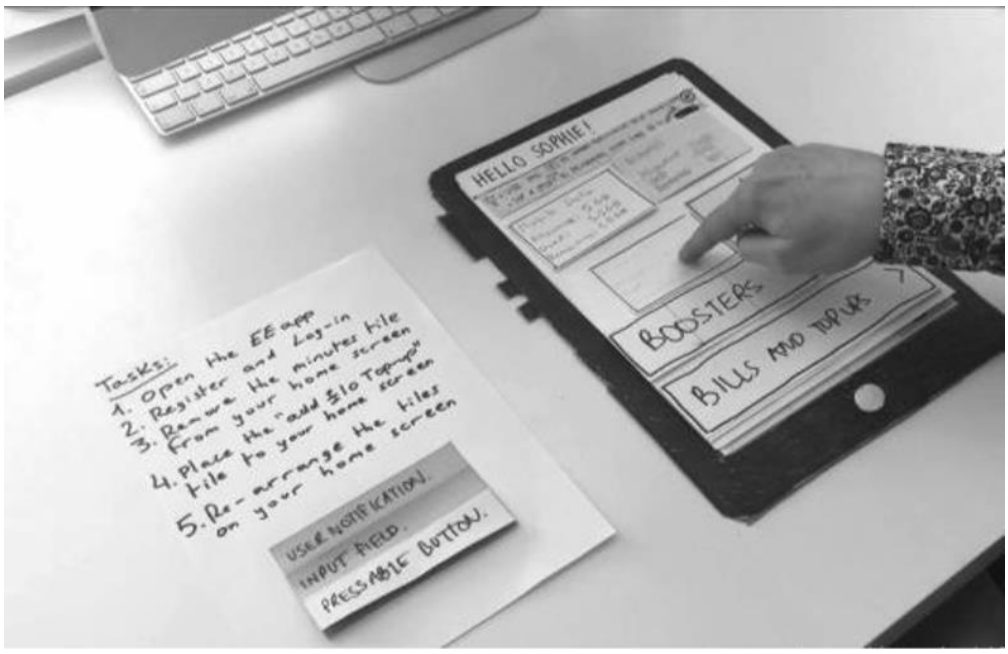
Name:_____

WEB DESIGN

LAYOUT

Database layout is designed using **wireframe diagrams** and **low-fidelity prototypes**.





At Advanced Higher level, these design techniques should also indicate the **underlying server-side processes** that will be executed.

SERVER-SIDE PROCESSES

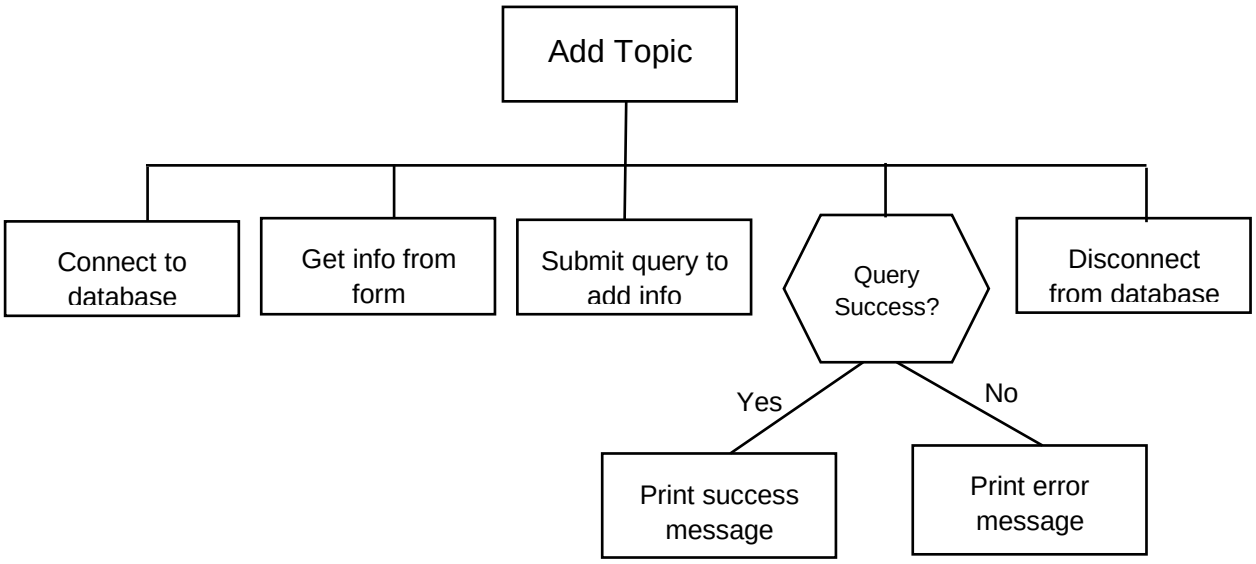
Server-side process are written in the PHP programming language. Examples of server-side processes include connecting to a database to execute SQL code or handling user login sessions.

Since these processes are written in code, they are designed using structure diagrams and pseudocode.

1. Add topic to database
2. Display topics from database

1.1 connect to database
1.2 get information from form (topic, detail, name, email, datetime
1.3 submit query to add information from form
1.4 if query is successful
1.5 print success message and a link to view topic
1.6 else
1.7 print error message
1.8 disconnect from database

2.1 connect to database
2.2 get information from database (SELECT * from db ORDER BY id DESC)
2.3 display ID,Topic, Views, Replies, Date/Time (with links to specific topic)
2.4 disconnect from database



IMPLEMENTATION:HTML

HTML TABLES

Tables are useful for laying out data in a web page. They are particularly useful when used to format the results of an SQL SELECT query to display the records and fields in an organized fashion.

An HTML table is defined with the **<table>** tag. Each table **row** is defined with the **<tr>** tag.

A table **heading** is defined with the **<th>** tag. By default, table headings are bold and centered. A table **data/cell** is defined with the **<td>** tag.

```
<table>
        <tr>
            <th>Firstname</th>     <th>Lastname</th>
        </tr>

        <tr>
           <td>Jill</td>      <td>Smith</td>
        </tr>

        <tr>
           <td>Eve</td>      <td>Jackson</td>
       </tr>
</table>
```

The above example would produce a table like the one below:

Firstname	**Lastname**
Jill	Smith
Eve	Jackson

To put borders around the table cells, a border style CSS rule like the one below should be used.

```
table, th, td {
  border: 1px solid black;
}
```

HTML FORMS

HTML forms are used to collect user input. The **<form>** element defines an HTML form:

```
<form>
    .
    form elements
    .
</form>
```

FORM ELEMENTS

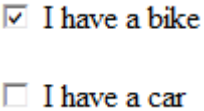
Forms make use of different types of element to allow the user to enter information.

Input

Input elements include are used for the most common ways in which a user will enter information into a form.

The type attribute defines the type of input.

`<input type="text" name="firstname" value="Mickey">`	
`<input type="number" name="age" value="25">`	
`<input type="password" name="psw">`	

`<input type="checkbox" name="vehicle1" value="Bike"> I have a bike`
 	
`<input type="checkbox" name="vehicle2" value="Car"> I have a car`
	

Input elements also include the submit button that the user can click on.

`<input type="submit" value="Submit">`	

Select

The Select element is used to define a drop-down list. This element is useful for restricted choice validation.

```
<select name="cars">
<option value="volvo">Volvo</option>
<option value="saab">Saab</option>
<option value="fiat">Fiat</option>
<option value="audi">Audi</option>
</select>
```

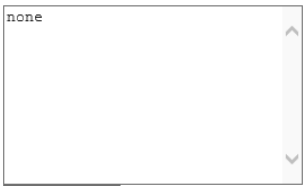
Textarea

The Textarea element is used to allow multi-line input for more extended pieces of information.

```
<textarea name="otherinfo" rows="10" cols="30">

      none

</textarea>
```



FORM VALIDATION

Form data should continue to be validated using the methods you learned about in Higher Computing Science.

Range Check	`<input type="number" name="score" min="1" max="6">`
Presence Check	`<input type="text" name="email" required>`
Length Check	`<input type="text" name="city" maxlength="15">`

SUBMITTING FORMS

The data entered into an HTML form is submitted to a web server for processing.

The submit input element is used to submit form data.

```
<form action="action_page.php" method="GET">
      <input type="submit" value="Submit">
</form>
```

Action Attribute

The **action attribute** defines the action to be performed when the form is submitted.

The action attribute contains the **name of the file** to be executed when the form is submitted. This can be the current file or a different file.

The common way to submit form data to a server, is by clicking on a submit button.

```
<form action="action_page.php" method="GET">
```

The action shown above states the file “action_page.php” will be opened when the form is submitted. As this is a PHP file, the web server where it is stored will automatically execute the PHP code contained in the file.

Method Attribute

The **method attribute** specifies the HTTP method to be used when submitting the forms.

The method used to submit the form can be either **GET** or **POST**:

- **GET** — the submitted data is visible to the user and therefore not secure
- **POST** — the submitted data is hidden from the user (forms are almost always submitted using the POST method)

GET

GET is usually used when **non-sensitive data** (like the parameters of a database query) is sent to a server.

If you submit a form with method="GET", the browser constructs a URL by taking the value of the action attribute, appending a **?** to it, then appending the form data set. It then processes this URL as if following a link.

The browser divides the URL into parts and recognises a host, then sends a GET request to that host, with the rest of the URL as an argument.

When you use GET, the form data will be visible in the page address:

```
action_page.php?firstname=Mickey&lastname=Mouse
```

Advantages of using GET:

- If security is not an issue, the URL can be bookmarked, allowing it to be re-used without having to complete and submit the original form.
- If there is a network connection issue when a form is submitted, the browser will automatically resend the form, as it assumes it does not contain sensitive data.
- GET submissions can usually be cached. If the same submission is used regularly (for example form data used to generate the same database query), this could have a significant effect on efficiency.

Disadvantages of using GET:

- The form data in the constructed URL is visible and so less secure.
- URLs can only contain ASCII codes, which will cause issues if the form data contains non-ASCII characters.
- The URL constructed will be stored in the user's web browsing history, making it inappropriate for sensitive data.
- URLs have a limited number of characters, which limits the form data submitted.

POST

POST is usually used when **sensitive data** (like personal information) is sent to a server. A database update would usually be initiated with the POST method.

POST can also be used for non-sensitive data: if the submitted data is likely to contain **non-ASCII characters** or the length is over the limit of a URL.

When you submit a form using the POST method, the form data set is encoded within a message that is sent to the server.If the form is updating data, or includes sensitive information (password).

Advantages of using POST:

- The submitted form data is not visible and so more secure than GET.
- Non-ASCII characters can be submitted within the form data set.
- There is no URL character limit, so form data can be much larger.

Disadvantages of using POST:

- The submitted form cannot be bookmarked for later use.
- If there is a network issue while the form is being submitted, the browser will ask the user to resubmit the form.

POST offers better security because the submitted data is not visible in the page address.

Key and Value Pairs

When the data in the form is submitted to the server, it is converted into a php array of key/value pairs.

- The **key** is the name of the form controls
- The **value** is the data entered by the user.

```
<input type="text" name="firstname" value="">
<input type="text" name="lastname" value="">
<input type="text" name="age" value="">
```

$_POST

Key	Value
firstname	Mickey
lastname	Mouse
Age	25

$_GET

Key	Value
firstname	Mickey
lastname	Mouse
Age	25

The name of the php array is either $_POST or $_GET depending on the method used.

In the case of a **drop-down menu** or a **radio button** input, both the name and value are defined in the HTML code.

```
Select play:
<select name="play">
  <option value="hamlet">Hamlet</option>
  <option value="godo">Waiting for Godo</option>
  <option value="brothers">Blood Brothers</option>
  <option value="curious">Curious Incident</option>
</select>
<br><br>
```

```
Choose your age:<br>
<input type="radio" name="age" value="12 to 14" checked> 12-14
<input type="radio" name="age" value="15 or 16"> 15,16
<input type="radio" name="age" value="over 16"> over 16
```

You will find more information on HTML forms at the following website:

http://www.w3schools.com/html/html_forms.asp