

Week 2 – Introduction to Web Design

Acknowledgement of Country

We respectfully acknowledge the Wurundjeri People of the Kulin Nation, who are the Traditional Owners of the land on which Swinburne's Australian campuses are located in Melbourne's east and outer-east, and pay our respect to their Elders past, present and emerging.

We are honoured to recognise our connection to Wurundjeri Country, history, culture, and spirituality through these locations, and strive to ensure that we operate in a manner that respects and honours the Elders and Ancestors of these lands.

We also respectfully acknowledge Swinburne's Aboriginal and Torres Strait Islander staff, students, alumni, partners and visitors.

We also acknowledge and respect the Traditional Owners of lands across Australia, their Elders, Ancestors, cultures, and heritage, and recognise the continuing sovereignties of all Aboriginal and Torres Strait Islander Nations.



Step 1 of Web Design: How to define requirements ?



Requirement definition is the work that needs to be done in our software development project for "project initiation management." The deliverables of the requirement definition phase have a direct impact on subsequent activities such as requirement gathering, analysis, and modeling, as well as development.

Requirement definition is the process of determining the high-level requirements of a project. In other words, it is about defining the business needs of the project, which involves clarifying the project's objectives and scope.

Step 1 of Web Design: How to define requirements ?

Requirement definition is the process of identifying the project's macro-level requirements. In other words, it involves defining the business needs of the project, which means clearly specifying the project's objectives and scope.

When to start defining requirements ?

Requirement definition should be addressed at the beginning of a project, specifically during project initiation. It should be completed during the project's inception.

Clear definition of project objectives and scope can guide the smooth progress of requirement gathering activities.

Step 1 of Web Design: How to define requirements ?

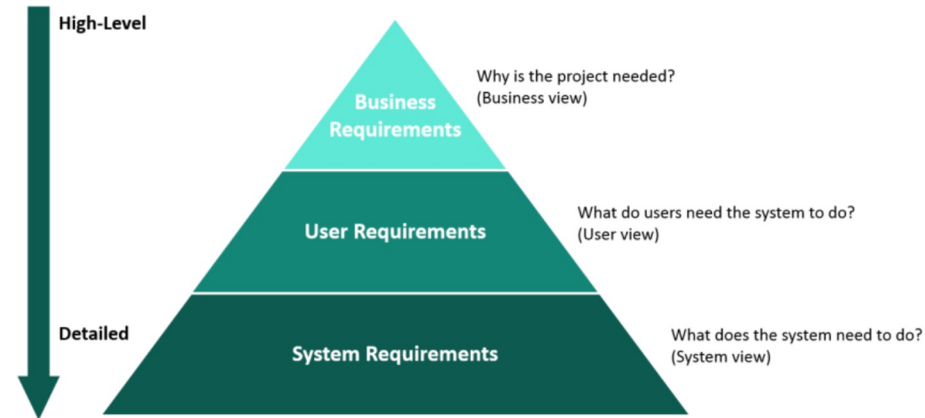
Deciphering ambiguous project objectives.

- Internal searching (trying to find project initiators within the company/organization) and having in-depth communication with these project initiators are the first steps in requirement definition.
- External tracing involves finding reference points to define requirements. Through analysis and understanding of these reference points, effective requirement definition can be achieved.

Step 1 of Web Design: How to define requirements ?

The following processing ideas can be used when defining requirements and producing a project proposal

Define the context and objectives of the project: Understand the origin and objectives of the project, and identify why the project was initiated and what it is expected to achieve.

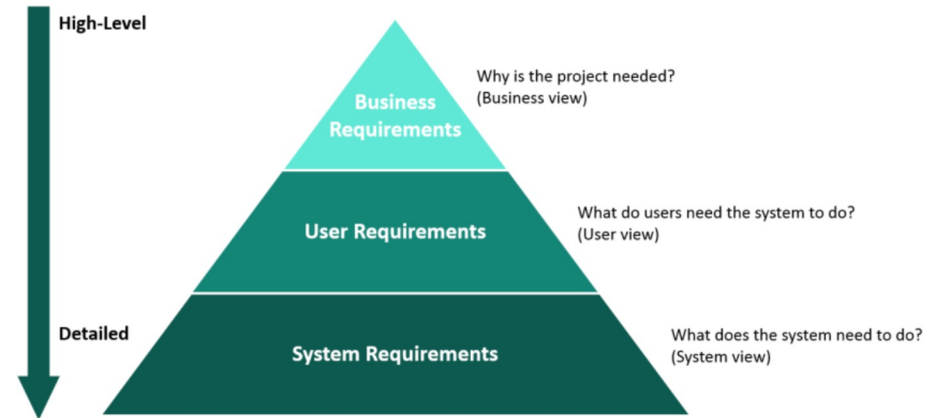


Requirements Hierarchy

Step 1 of Web Design: How to define requirements ?

The following processing ideas can be used when defining requirements and producing a project proposal

Gather stakeholder requirements:
Communicate with various stakeholders to understand their needs and expectations, and ensure that the project fully considers the requirements of all key stakeholders.



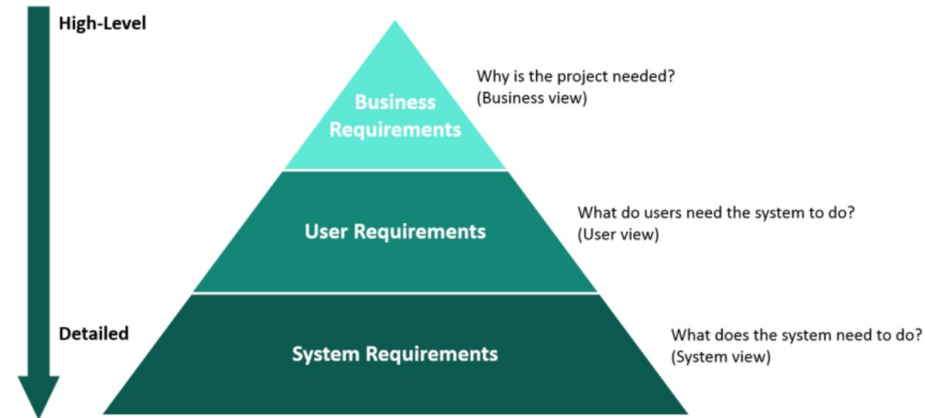
Requirements Hierarchy

Image from CRVS Digitisation Guidebook

Step 1 of Web Design: How to define requirements ?

The following processing ideas can be used when defining requirements and producing a project proposal

Analyse the scope of the project: Define the boundaries of the project, including the functions to be covered, the target users, the geographic scope, etc.



Requirements Hierarchy

Step 2 of Web Design: Why Prototype Design ?

Prototyping plays a crucial role in projects for several reasons:

- Clarify Requirements: Prototyping helps teams and stakeholders better understand project requirements. Through visualisation, prototyping allows everyone to have a clearer understanding of the project's functionality and interfaces, avoiding confusion and misunderstanding in the understanding of requirements.



Image from uxp.in.com

Step 2 of Web Design: Why Prototype Design ?

Prototyping plays a crucial role in projects for several reasons:

- Identify problems in advance: Through prototyping, project teams can identify and solve potential problems before development. This helps to reduce the cost of modifications and adjustments at a later stage of the project, while increasing the likelihood of project success.



Image from uxp.in.com

Step 2 of Web Design: Why Prototype Design ?

Prototyping plays a crucial role in projects for several reasons:

- Effective communication: Prototyping as a communication tool can help the development team, designers and stakeholders communicate more effectively with each other. It can help different roles understand and unify their understanding of the project and reduce communication errors.



Image from uxp.in.com

Step 2 of Web Design: Why Prototype Design ?

Prototyping plays a crucial role in projects for several reasons:

- Ensure consistency: prototyping ensures that the entire team is aligned on the goals and style of the project. This is especially important in large projects or when multiple people are working together to avoid disagreements between multiple team members.



Image from uxp.in.com

Step 2 of Web Design: Why Prototype Design ?

Prototyping plays a crucial role in projects for several reasons:

- Ensure consistency: prototyping ensures that the entire team is aligned on the goals and style of the project. This is especially important in large projects or when multiple people are working together to avoid disagreements between multiple team members.



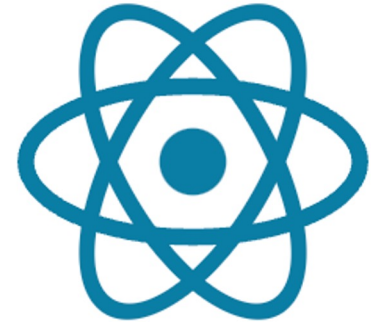
Image from uxp.in.com

Step 3 of Web Design: Building Your Web with React.js



React.js is a popular JavaScript library developed and maintained by Facebook that has many features that make it ideal for building modern front-end applications. Here are the key features of React.js:

- Componentised development: React.js uses a componentised development model to split the application into multiple independent, reusable components. Each component has its own state and behaviour and can be nested and combined.
- Virtual DOM: React.js introduces the concept of a virtual DOM, which is a lightweight copy of the actual DOM similar to the browser's. When data changes, React improves rendering performance and efficiency by comparing the differences between the virtual DOM and the actual DOM and minimally updating the actual DOM.



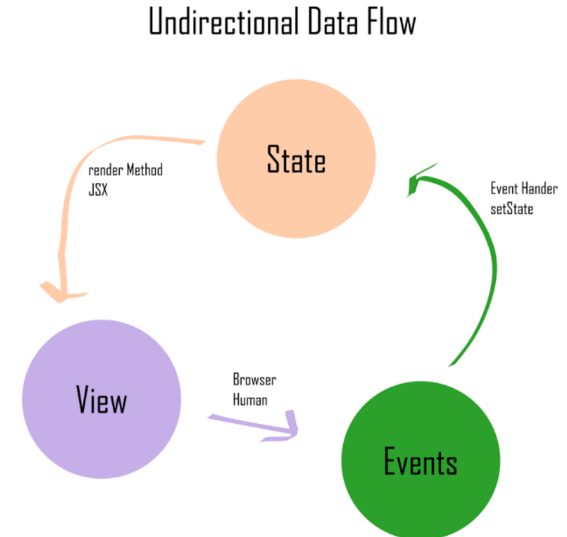
React

Step 3 of Web Design: Building Your Web with React.js



React.js is a popular JavaScript library developed and maintained by Facebook that has many features that make it ideal for building modern front-end applications. Here are the key features of React.js:

- Unidirectional data flow: React.js follows the principle of unidirectional data flow, where data is passed from parent component to child component.



Step 3 of Web Design: Building Your Web with React.js



React Playground: playcode.io/react

A screenshot of the React Playground web application. The interface is dark-themed. On the left, a 'FILES' sidebar shows a project structure with 'src' containing 'App.jsx', 'index.jsx', and 'style.css', and 'package.json' at the root. Below this, 'PACKAGES' lists 'react' and 'react-dom'. The main editor area shows the code in 'App.jsx':

```
1 import React from 'react';
2
3 export function App(props) {
4   return (
5     <div className='App'>
6       <h1>Hello React.</h1>
7       <h2>Start editing to see some magic happen!</h2>
8     </div>
9   );
10 }
11
12 // Log to console
13 console.log('Hello console')
```

The bottom of the interface is split into two panels. The 'Console' panel on the left shows the output 'Hello console'. The 'Web View' panel on the right displays the rendered HTML: 'Hello React.' followed by 'Start editing to see some magic happen!' in a light green font. A small help icon (?) is visible in the bottom right corner.

Step 3 of Web Design: Building Your Web with React.js



In React, **import** and **from** are JavaScript keywords used to import modules, which are used to bring in modules, components, functions, or variables defined in other files. These keywords are typically used with ES6's module system.

import: The import keyword is used to import exported content from other modules. It can bring in either a named export or a default export. An example is shown below:

JavaScript ▾

```
import { Component1, Component2, variable1 } from './components';  
  
import DefaultComponent from './default-component';
```

Step 3 of Web Design: Building Your Web with React.js



In React, **import** and **from** are JavaScript keywords used to import modules, which are used to bring in modules, components, functions, or variables defined in other files. These keywords are typically used with ES6's module system.

from: The from keyword is used to specify the module from which to import content. It immediately follows the import keyword. After from, we specify the relative or absolute path to the module to tell the JavaScript engine from which file to get the module.

In React, it's common to use these keywords in component files to import other components, tool functions, styles, or other dependencies. Such modular development makes code more organised and easier to maintain

JavaScript ▾

```
import { Component1, Component2, variable1 } from './components';  
  
import DefaultComponent from './default-component';
```

Step 3 of Web Design: Building Your Web with React.js



In JavaScript, **class** and **extends** are keywords used to create and inherit classes.

class: The class keyword is used to create a class. A class is an object constructor that defines a set of objects that share the same characteristics and methods. In a class, we can define constructors, member variables, member methods, etc.

```
class Test {  
  constructor(param1, param2) {  
    this.param1 = param1;  
    this.param2 = param2;  
  }  
  
  helloWorld() {  
    console.log(`param1 is ${this.param1} and param2 is ${this.param2}`);  
  }  
}  
  
const test1 = new Test('hello', 30);  
test1.helloWorld(); // Output: param1 is hello and param2 is 30
```

Step 3 of Web Design: Building Your Web with React.js



In JavaScript, **class** and **extends** are keywords used to create and inherit classes.

extends: The extends keyword is used to inherit from one class in another class. With the extends keyword, a subclass can inherit the attributes and methods of the parent class and also have its own unique attributes and methods.

```
class Test2 extends Test {
  constructor(param1, param2, param3) {
    super(param1, param2); // use super function call constructor in parent class
    this.param3 = param3;
  }

  test2Hello() {
    console.log(`param1-${this.param1},param2 ${this.param2},param3-${this.param3}`);
  }
}

const student1 = new Student('hello', 30, 40);
student1.test2Hello(); // Output: param1-hello,param2 30,param3-40`
student1.helloWorld(); // Output: param1 is hello and param2 is 30`
```

Step 3 of Web Design: Building Your Web with React.js



In React, the **render** function is a required method to define the output of a component. Every React component must contain a render function that is responsible for returning a React element or a set of React elements that describe the output of the component on the screen.

The main purpose of the render function is to generate a virtual DOM based on the component's current state (state) and properties (props) and render it into the actual DOM.

index.jsx × App.jsx ×

```
1 import React from 'react';
2 import ReactDOM from 'react-dom/client';
3
4 import { App } from './App.jsx'
5
6 ReactDOM.createRoot(
7   document.querySelector('#root')
8 ).render(<App />)
```

index.jsx × App.jsx ×

```
1 import React from 'react';
2
3 export function App(props) {
4   return (
5     <div className='App'>
6       <h1>Hello React.</h1>
7       <h2>Start editing to see some magic happen!</h2>
8     </div>
9   );
10 }
11
12 // Log to console
13 console.log('Hello console!')
```

Step 3 of Web Design: Building Your Web with React.js



In React, the **render** function is a required method to define the output of a component. Every React component must contain a render function that is responsible for returning a React element or a set of React elements that describe the output of the component on the screen.

The main purpose of the render function is to generate a virtual DOM based on the component's current state (state) and properties (props) and render it into the actual DOM.

index.jsx × App.jsx ×

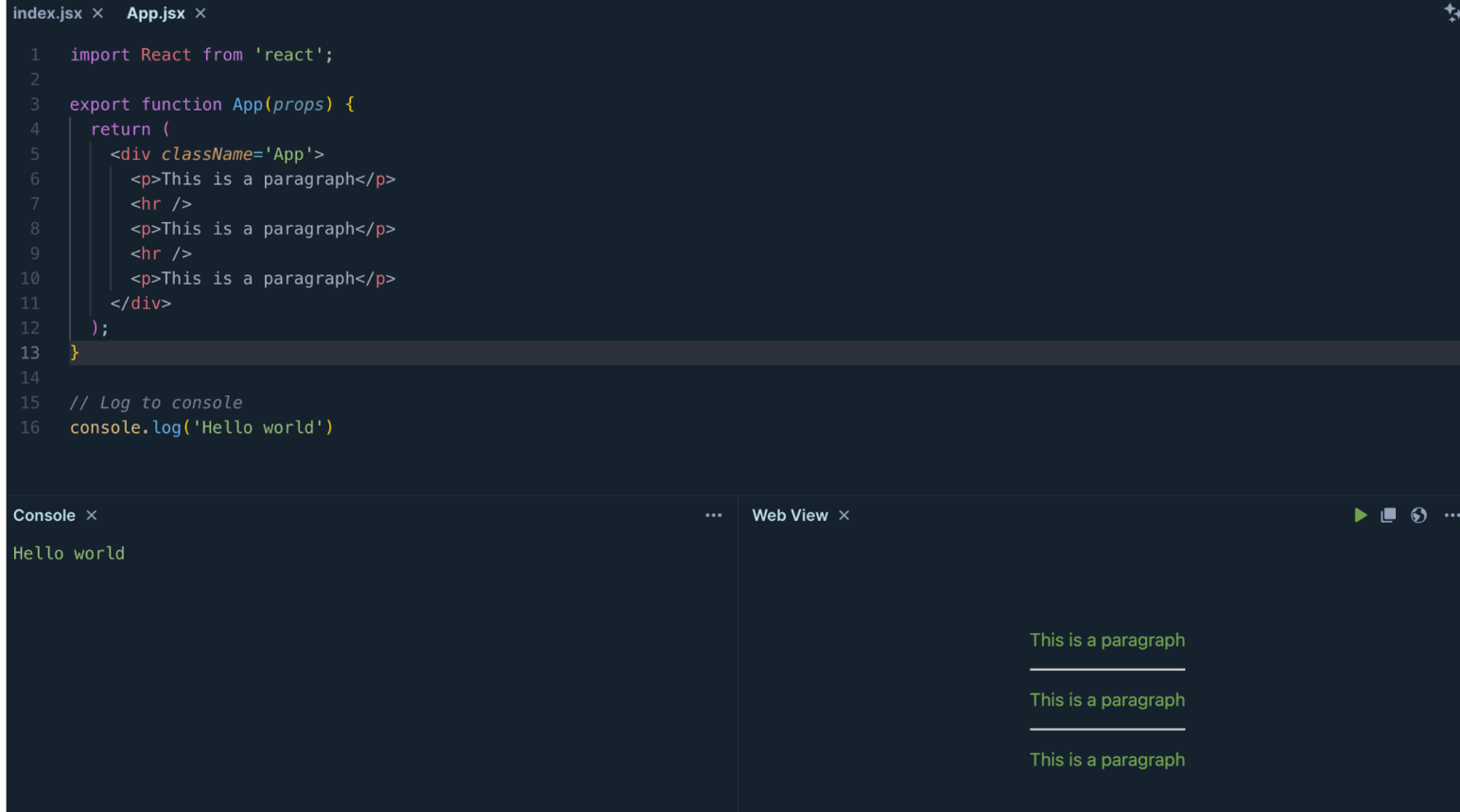
```
1 import React from 'react';
2 import ReactDOM from 'react-dom/client';
3
4 import { App } from './App.jsx'
5
6 ReactDOM.createRoot(
7   document.querySelector('#root')
8 ).render(<App />)
```

index.jsx × App.jsx ×

```
1 import React from 'react';
2
3 export function App(props) {
4   return (
5     <div className='App'>
6       <h1>Hello React.</h1>
7       <h2>Start editing to see some magic happen!</h2>
8     </div>
9   );
10 }
11
12 // Log to console
13 console.log('Hello console!')
```

Step 3 of Web Design: Integrating HTML in React.js

In React, HTML elements are also support in the React.js

A screenshot of a code editor with a dark theme. The top pane shows the code for App.jsx, which imports React and defines a function App that returns a JSX element containing three paragraphs separated by horizontal lines. The bottom pane is split into a Console and a Web View. The Console shows the log output 'Hello world'. The Web View shows the rendered HTML output, which displays three paragraphs of text, each followed by a horizontal line.

```
index.jsx x App.jsx x
1  import React from 'react';
2
3  export function App(props) {
4    return (
5      <div className='App'>
6        <p>This is a paragraph</p>
7        <hr />
8        <p>This is a paragraph</p>
9        <hr />
10       <p>This is a paragraph</p>
11     </div>
12   );
13 }
14
15 // Log to console
16 console.log('Hello world')
```

Console x ... Web View x

Hello world

This is a paragraph

This is a paragraph

This is a paragraph

Step 3 of Web Design: Integrating HTML in React.js

In React, HTML elements are also support in the React.js

index.jsx × App.jsx ×

```
1 import React from 'react';
2
3 export function App(props) {
4   return (
5     <div className='App'>
6       <h1>This is a heading</h1>
7       <h2>This is a heading</h2>
8       <h3>This is a heading</h3>
9     </div>
10   );
11 }
12
13 // Log to console
14 console.log('Hello world')
```

Console ×

Hello world

Web View ×

This is a heading

This is a heading

This is a heading

Step 3 of Web Design: Integrating HTML in React.js

In React, HTML elements are also support in the React.js

```
index.jsx x App.jsx x
1  import React from 'react';
2
3  export function App(props) {
4    return (
5      <div className='App'>
6        <ul>
7          <li><a href="#home">home</a></li>
8          <li><a href="#news">news</a></li>
9          <li><a href="#contact">contact</a></li>
10         <li><a href="#about">about</a></li>
11        </ul>
12      </div>
13    );
14  }
15
16  // Log to console
17  console.log('Hello world')
```

Console x ... Web View x

Hello world

- [home](#)
- [news](#)
- [contact](#)
- [about](#)

Learning Resources



How to use Axure as the prototype design tool

<https://docs.axure.com/axure-rp/reference/getting-started-video/>

Define Your Project Requirements: 11 Steps

<https://www.bairesdev.com/blog/define-your-project-requirements/>

React Playground

<https://playcode.io/react>

HTML Tag

https://www.w3schools.com/tags/tag_html.asp

