

## P1

The screenshot shows a browser window with several tabs open at the top: '記事 - Evernote', '[1101-1N] Connect Github Using Vercel', 'New Project – Vercel', 'G z翻譯 - Google 搜尋', and 'WeiYiHuangFelix/1102-javascript-210410...'.

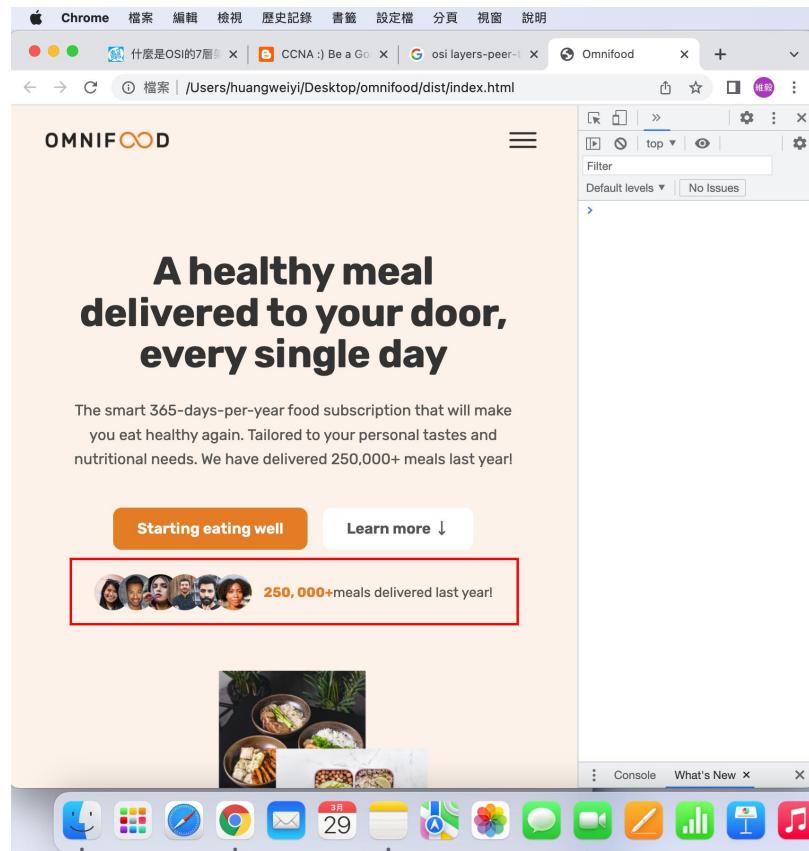
The main content area displays a 'Congratulations!' message: 'You just deployed a new Project to Vercel.' Below this, there's a preview of the 'OMNIFOOD' website, which features a banner about a healthy meal delivery service, a photo of a woman eating, and a call-to-action button 'Starting eating well'.

To the right of the preview, there are three sections: 'Go to Dashboard', 'DEVELOP' (instructions to run locally), and 'SHIP' (instructions to push to production). Below these is a 'PREVIEW' section with instructions to push to a Git branch other than 'main' to preview changes.

The screenshot shows a GitHub repository page for 'WeiYiHuangFelix/1102-javascript-210410626'. The repository is private. The main navigation bar includes 'Search or jump to...', 'Pull requests', 'Issues', 'Marketplace', and 'Explore'.

The repository details section shows the following information:

- Code:** main (branch), 1 branch, 0 tags
- Commits:**
  - WeiYiHuangFelix initial commit (da9d1ac, 24 minutes ago)
  - demo/omnifood Initial commit (24 minutes ago)
- Add a README:** A button to add a README file.
- About:** No description, website, or topics provided.
- Statistics:** 0 stars, 1 watching, 0 forks.
- Releases:** No releases published. Create a new release.
- Packages:** No packages published. Publish your first package.



The screenshot shows a meal delivery service website. At the top, there's a navigation bar with tabs like '什麼是OSI的7層?' and 'CCNA : Be a Go...'. Below the navigation is a search bar and a sidebar with a 'Default levels' dropdown. The main content features a large headline: 'A healthy meal delivered to your door, every single day'. Below the headline is a subtext: 'The smart 365-days-per-year food subscription that will make you eat healthy again. Tailored to your personal tastes and nutritional needs. We have delivered 250,000+ meals last year!'. There are two buttons: 'Starting eating well' and 'Learn more ↓'. A red box highlights a section with a grid of small profile pictures and the text '250,000+meals delivered last year!'. Below this is a thumbnail image of several prepared meals.

```

const imgs = [
  {
    img: "./img/customers/customer-1.jpg",
    img1: "./img/customers/customer-2.jpg",
    img2: "./img/customers/customer-3.jpg",
    img3: "./img/customers/customer-4.jpg",
    img4: "./img/customers/customer-5.jpg",
    img5: "./img/customers/customer-6.jpg",
    span: "250, 000+",
    p: "meals delivered last year!",
  },
];

const deliveredMeals = document.querySelector('.delivered-meals');

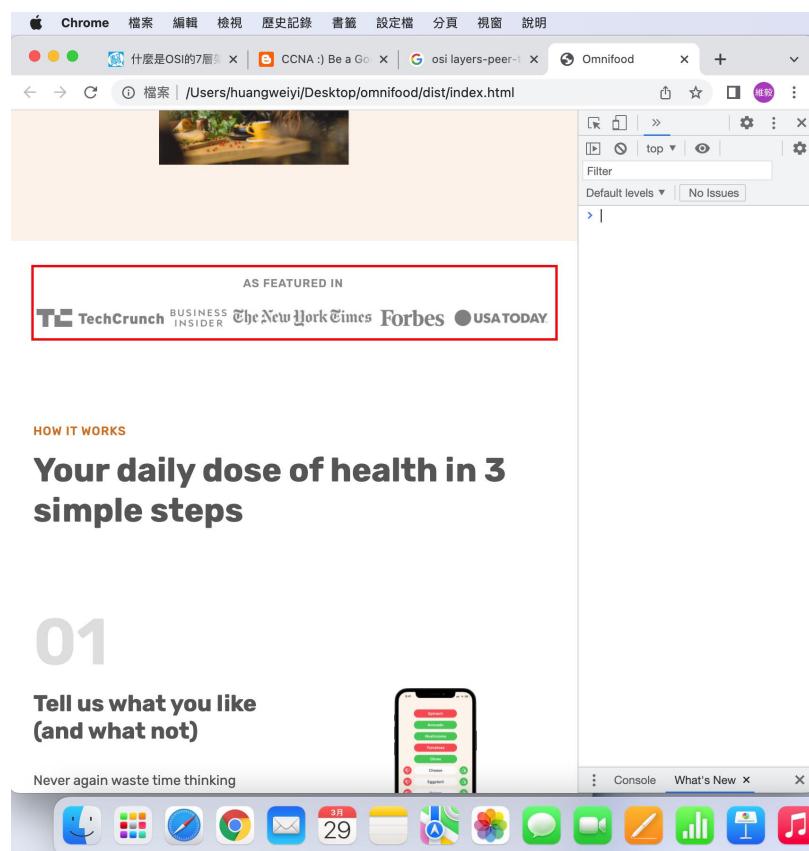
window.addEventListener('DOMContentLoaded', () => {
  | displayImgItems(imgs);
});

const displayImgItems = (imgsItems) => {
  let displayImgs = imgsItems.map((item) => {
    //console.log('item', item);

    return `
      <div class="delivered-imgs">
        <img src=${item.img} alt="Customer photo" />
        <img src=${item.img1} alt="Customer photo" />
        <img src=${item.img2} alt="Customer photo" />
        <img src=${item.img3} alt="Customer photo" />
        <img src=${item.img4} alt="Customer photo" />
        <img src=${item.img5} alt="Customer photo" />
    `;
  });

  return displayImgs;
}

```



The screenshot shows a meal delivery service website. At the top, there's a navigation bar with tabs like '什麼是OSI的7層?' and 'CCNA : Be a Go...'. Below the navigation is a search bar and a sidebar with a 'Default levels' dropdown. The main content features a large image of a meal. Below the image is a red box containing logos for 'TechCrunch', 'Business Insider', 'The New York Times', 'Forbes', and 'USA Today'. The text 'AS FEATURED IN' is above the logos. Below this is a section titled 'HOW IT WORKS' with the sub-section 'Your daily dose of health in 3 simple steps'. Step 1 is labeled '01' and has the sub-section 'Tell us what you like (and what not)'. It includes an image of a smartphone displaying a meal selection interface. The bottom of the page has a footer with the text 'Never again waste time thinking'.

```

const logos = [
  {
    img: "img/logos/techcrunch.png",
    img1: "img/logos/business-insider.png",
    img2: "img/logos/the-new-york-times.png",
    img3: "img/logos/forbes.png",
    img4: "img/logos/usa-today.png",
  },
];

const sectionFeatured = document.querySelector('.section-featured');

window.addEventListener('DOMContentLoaded', () => {
  | displayLogosItems(logos);
});

const displayLogosItems = (logosItems) => {
  let displayLogos = logosItems.map((item) => {
    //console.log('item', item);

    return `
      <div class="container">
        <h2 class="heading-featured-in">As featured in</h2>
        <div class="logos">
          <img src=${item.img} alt="Techcrunch logo" />
          <img src=${item.img1} alt="Business Insider logo" />
          <img src=${item.img2} alt="The New York Times logo" />
          <img src=${item.img3} alt="Forbes logo" />
          <img src=${item.img4} alt="USA Today logo" />
        </div>
      </div>
    `;
  });

  return displayLogos;
}

```

The screenshot shows a Mac desktop with two browser windows open. The left window displays the `script.js` file, which contains JavaScript code for displaying gallery items. The right window shows the `index.html` page, which features a grid of food images and some testimonial text. The developer tools are open in both windows, highlighting specific sections of the code and the resulting DOM structure.

```

const galleries = [
  {
    img: "img/gallery/gallery-1.jpg",
    img1: "img/gallery/gallery-2.jpg",
    img2: "img/gallery/gallery-3.jpg",
    img3: "img/gallery/gallery-4.jpg",
    img4: "img/gallery/gallery-5.jpg",
    img5: "img/gallery/gallery-6.jpg",
    img6: "img/gallery/gallery-7.jpg",
    img7: "img/gallery/gallery-8.jpg",
    img8: "img/gallery/gallery-9.jpg",
    img9: "img/gallery/gallery-10.jpg",
    img10: "img/gallery/gallery-11.jpg",
    img11: "img/gallery/gallery-12.jpg"
  },
};

const gallery = document.querySelector('.gallery');

window.addEventListener('DOMContentLoaded', () => {
  displayGalleryItems(gallery);
});

const displayGalleryItems = (galleriesItems) => {
  let displayGalleries = galleriesItems.map((item) => {
    console.log('item', item);

    return `
      <figure class="gallery-item">
        <img
          src=${item.img}
          alt="Photo of beautifully arranged food"
        />
        <!-- <figcaption>Caption</figcaption> -->
      </figure>
      <figure class="gallery-item">
        <img
          alt="Photo of a chef preparing food"
        />
      </figure>
    `;
  });

  return displayGalleries;
}

document.querySelector('.gallery').innerHTML = displayGalleries();

```

The screenshot shows a Mac desktop with two browser windows open. The left window displays the `script.js` file, which contains JavaScript code for displaying testimonial items. The right window shows the `index.html` page, which features a grid of testimonial cards. The developer tools are open in both windows, highlighting specific sections of the code and the resulting DOM structure.

```

const testimonial = [
  {
    img: "img/customers/dave.jpg",
   blockquote: "Inexpensive, healthy and great-tasting meals, without even having to order manually! It feels truly magical.",
    p: "Dave Bryson",
    img1: "img/customers/ben.jpg",
    blockquote1: "The AI algorithm is crazy good, it chooses the right meals for me every time. It's amazing not to worry about food anymore!",
    p1: "Ben Hadley",
    img2: "img/customers/steve.jpg",
    blockquote2: "Omnifood is a life saver! I just started a company, so there's no time for cooking. I couldn't live without my daily meals now!",
    p2: "Steve Miller",
    img3: "img/customers/hannah.jpg",
    blockquote3: "I got Omnifood for the whole family, and it frees up so much time! Plus, everything is organic and vegan and without plastic.",
    p3: "Hannah Smith",
  },
];

const testimonials = document.querySelector('.testimonials');

window.addEventListener('DOMContentLoaded', () => {
  displayTestimonialItems(testimonials);
});

const displayTestimonialItems = (testimonialItems) => {
  let displayTestimonial = testimonialItems.map((item) => {
    console.log('item', item);

    return `
      <figure class="testimonial">
        <img
          alt="Photo of a customer testimonial"
        />
        <p>${item.blockquote}</p>
        <p>${item.p}</p>
        <img
          alt="Photo of another customer testimonial"
        />
        <p>${item.blockquote1}</p>
        <p>${item.p1}</p>
        <img
          alt="Photo of a third customer testimonial"
        />
        <p>${item.blockquote2}</p>
        <p>${item.p2}</p>
        <img
          alt="Photo of a fourth customer testimonial"
        />
        <p>${item.blockquote3}</p>
        <p>${item.p3}</p>
      </figure>
    `;
  });

  return displayTestimonial;
}

document.querySelector('.testimonials').innerHTML = displayTestimonial();

```

## 綜合學習甘苦談

目前有重複看教授的影片，來了解各種方式的寫法，只是目前可能還是無法讓自己能馬上想出要如何自己寫，還是需要參考，才能慢慢寫出來

## 如何完成

一開始真的不會寫，只能模仿教授上課所教的，去運用理解程式碼的意思寫出來，如果發現錯誤就用 `log` 方式顯示出來 看哪裡有錯  
誤再修正，一步一步完成

## 過程中 解決哪先困擾問題

第一個就是變數名稱，如果沒有改變的話，在這句就會顯示錯誤，無法顯示，所以變數名稱不要相同 `const sectionFeatured = document.querySelector('.section-featured');`

第二個 `return` 裡面如果變數名稱設錯，就無法有效的呈現陣列裡面的值，造成畫面無法顯示出來

第三個 一定要 `console.log` 出來，不管對或錯，常常檢視 `log` 是讓自己更容易看懂程式碼錯誤與否

## 自主學習

我自主學習是看網路 youtube 布魯斯前端,學習基礎 加深印象 了解更多 javascript 使用方法，屬性函式方法的運用

第二點是看自己買的 javascript 書籍來閱讀，讓自己可以加快吸收知識