

Contenido

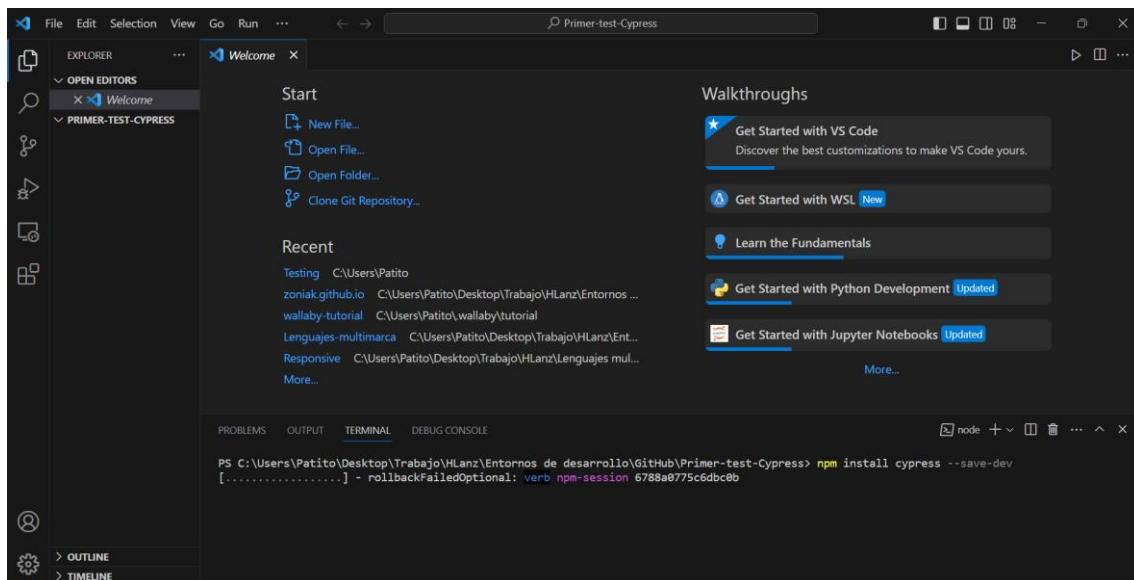
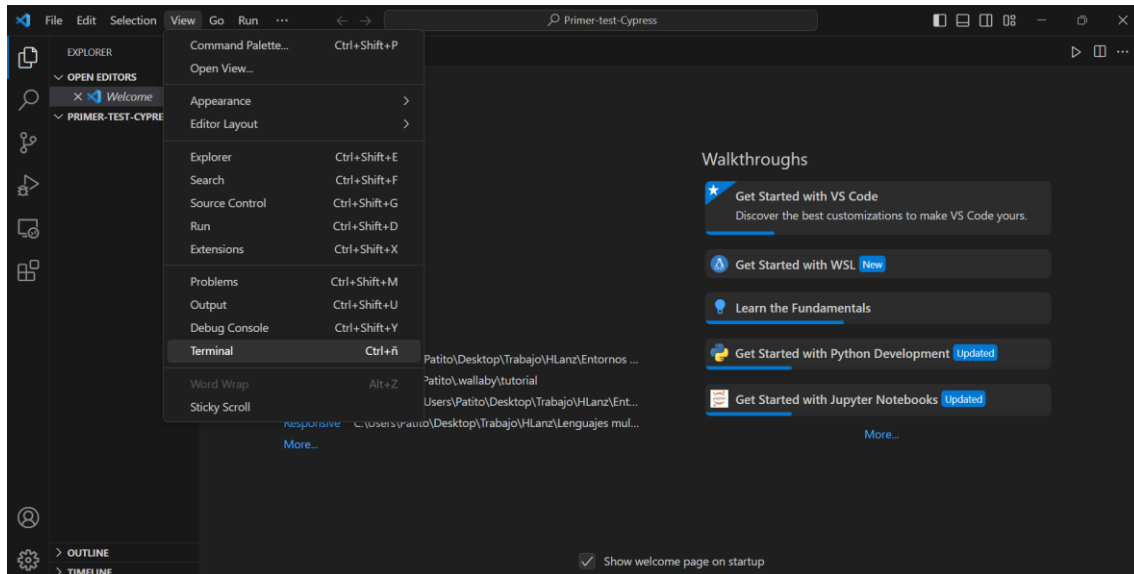
1.	Creación del primer test con Cypress	2
1.1.	Elementos necesarios para la practica.....	2
1.2.	Empezando a escribir el primer test	3
2.	Uso de test reales con Cypress	8
2.1.	Requisitos previos.....	8
2.2.	Fork de la aplicación	8
2.3.	Lanzamiento de la aplicación	9

1. CREACIÓN DEL PRIMER TEST CON CYPRESS

1.1. ELEMENTOS NECESARIOS PARA LA PRACTICA

Lo primero de todo va a ser tener instalado nvm para Windows. Si aun no lo tienes te dejo este [link](#) donde puedes encontrar los archivos necesarios para instalarlo. Para poder usar npm vamos a tener que hacer `nvm install latest` y después `nvm use latest`. Una vez lo tengamos, navegamos hasta la carpeta donde este el proyecto (con CMD y cd o simplemente abriendo el terminal en el proyecto en Visual Studio Code), y procedemos a instalar Cypress con el siguiente comando:

npm install cypress --save-dev



Algo curioso que os puede venir bien para el futuro es que con nvm podemos crear la estructura de un proyecto, para ello vamos a ejecutar el siguiente comando:

npm init vite

Al ejecutar esto nos pedirá datos como el nombre del proyecto o qué tipo de proyecto queremos crear:

```
PROBLEMS  OUTPUT  TERMINAL  DEBUG CONSOLE

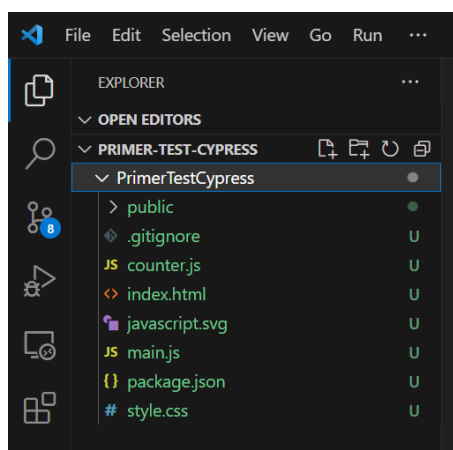
✓ Project name: ... PrimerTestCypress
✓ Package name: ... primertestcypress
? Select a framework: » - Use arrow-keys. Return to submit.
> Vanilla
  Vue
  React
  Preact
  Lit
  Svelte
```

Vamos a crear un proyecto Vanilla, y vamos a elegir como lenguaje JavaScript:

```
PROBLEMS  OUTPUT  TERMINAL  DEBUG CONSOLE

✓ Select a framework: » Vanilla
? Select a variant: » - Use arrow-keys. Return to submit.
  TypeScript
> JavaScript
```

Ahora ya tenemos nuestro proyecto creado:

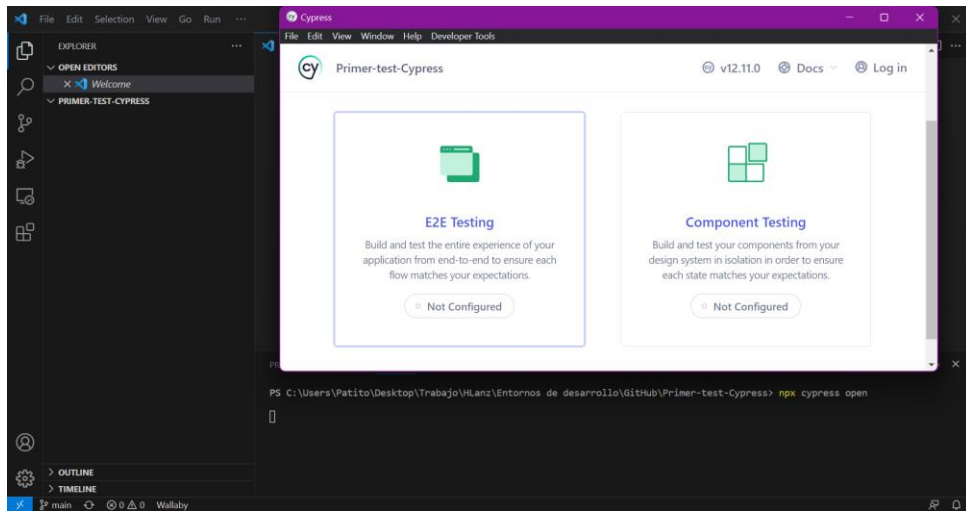


1.2. EMPEZANDO A ESCRIBIR EL PRIMER TEST

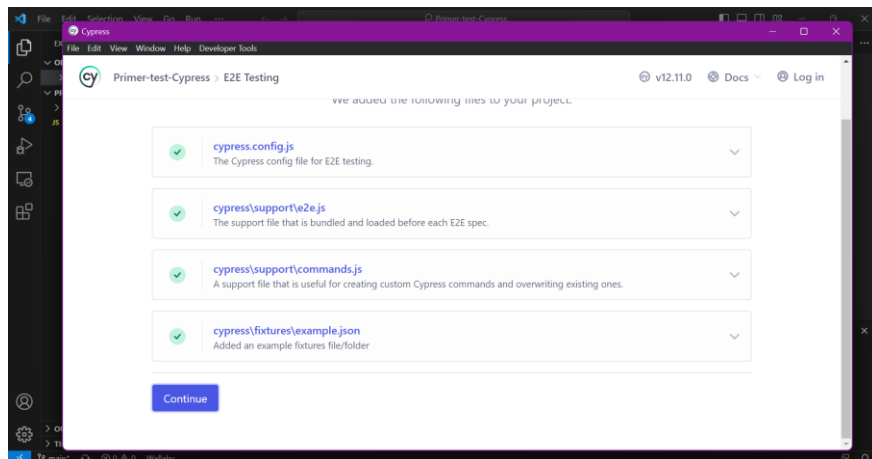
Vale ahora que tenemos instalado nvm y Cypress vamos a lanzar nuestro primer test. Para ello creamos un repositorio en GitHub para esta clase y empezamos a trabajar en el en Visual Studio Code. En el terminal ejecutamos este comando:

`npx cypress open`

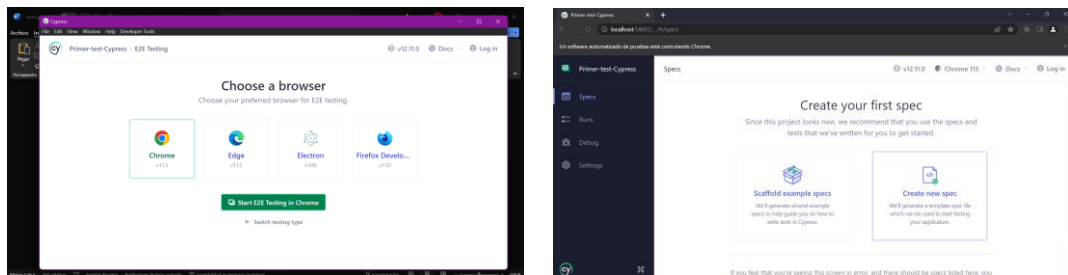
Se nos abrirá la aplicación de Cypress, vamos a crear el entorno para test E2E:



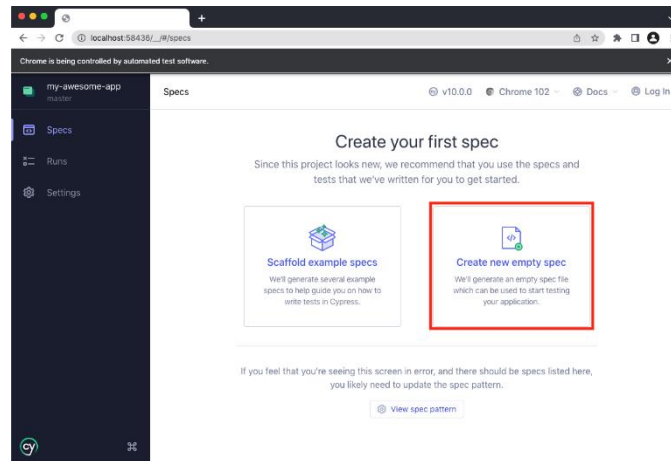
Le damos a continue:



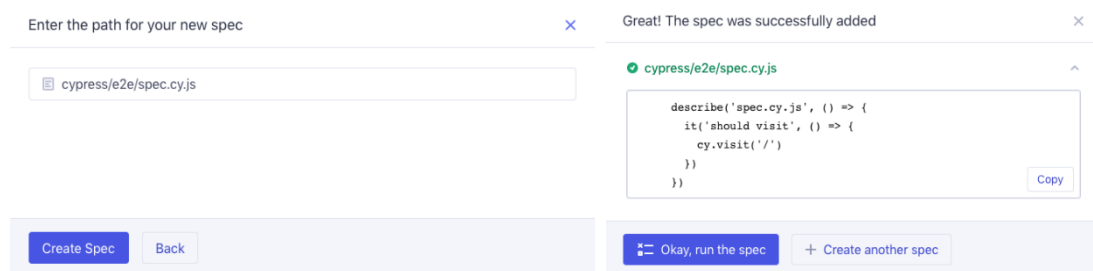
Y ejecutamos el programa en el navegador:



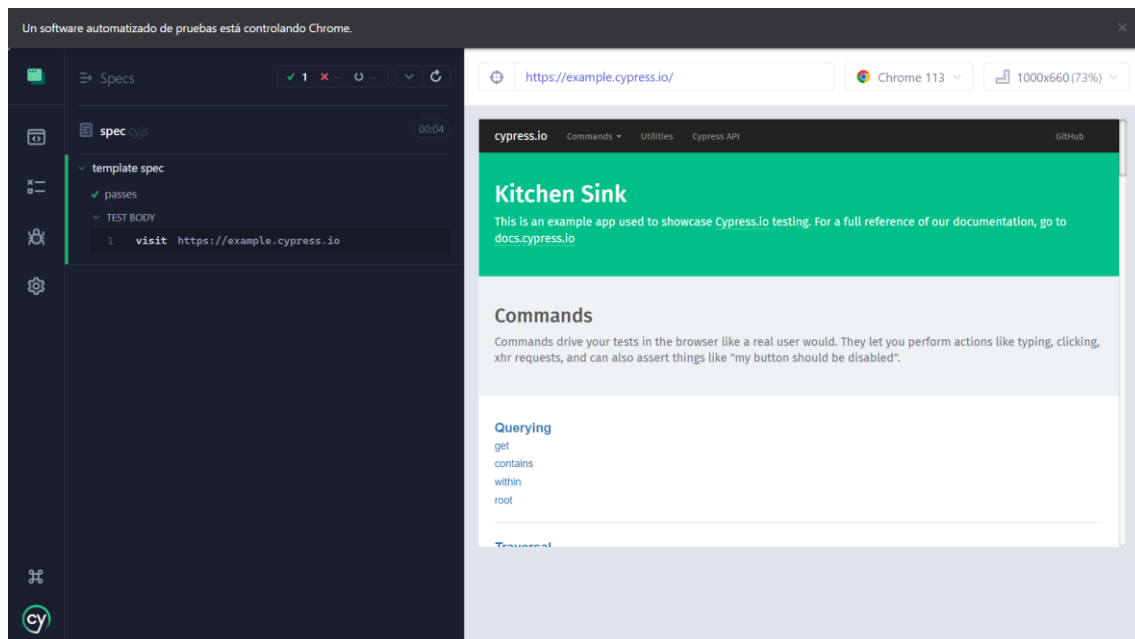
Ahora vamos a crear el primer test para comprobar que funciona. Para eso primero hacemos click en *Create new spec*



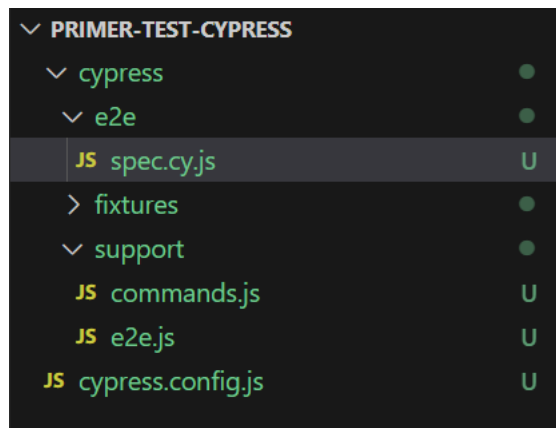
Procedemos a dejar todo como viene por defecto:



Y a continuación se abrirá este dónde se ejecutará y nos dará un resultado:



Ahora vamos a escribir nosotros nuestras pruebas. Para ello buscamos el archivo que se ha creado el cual es spec.cy.js que se encuentra en la siguiente ruta:



Vamos a escribir una aplicación simple con la que testear el código. Lo primero es crear las siguientes funciones:

```
//-- Dummy application code
let add = (a,b) => a+b;
let subtract = (a,b) => a-b;
let divide = (a,b) => a/b;
let multiply = (a,b) => a*b;
```

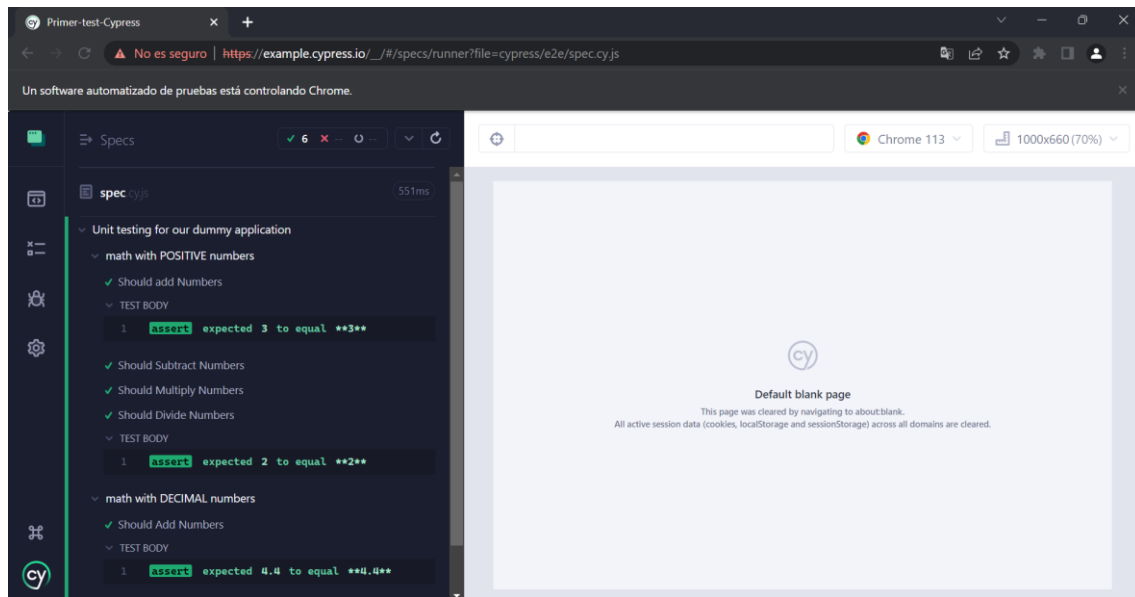
[Si pinchas en la imagen te llevara al código](#)

Después vamos a preparar los test, y para esto vamos a usar **describe**. Este es un método que nos sirve para agrupar en Cypress uno o más test relacionados. Procedemos a escribir el siguiente código:

```
describe('Unit testing for our dummy application', () => {
  context('math with POSITIVE numbers', () => {
    it('Should add Numbers', () => {
      expect(add(1,2)).to.eq(3);
    });
    it('Should Subtract Numbers', () => {
      expect(subtract(2,1)).to.eq(1);
    });
    it('Should Multiply Numbers', () => {
      expect(multiply(2,2)).to.eq(4);
    });
    it('Should Divide Numbers', () => {
      expect(divide(4,2)).to.eq(2);
    });
  });
  context('math with DECIMAL numbers', () => {
    it('Should Add Numbers', () => {
      expect(add(2.2,2.2)).to.eq(4.4);
    });
    it('Should Subtract Numbers', () => {
      expect(subtract(4.4, 2.2)).to.eq(2.2);
    });
  });
});
```

[Si pinchas en la imagen te llevara al código](#)

Según los vayamos escribiendo el navegador se ira actualizando automáticamente:



Ahora es vuestro turno. Mirar en la [documentación de Cypress](#) distintos tipos de test y probar estos. Por ejemplo con el siguiente código va a visitar una pagina y hacer click :

```
describe('My First Test', () => {
  it('clicks the link "type"', () => {
    cy.visit('https://example.cypress.io')
    cy.contains('type').click()
  })
})
let multiply = (a,b) => a*b;
```

Yo he probado a ejecutarlo con mi portfolio y me ha dado errores que no esperaba...

2. USO DE TEST REALES CON CYPRESS

2.1. REQUISITOS PREVIOS.

Lo primero que vamos a necesitar (partiendo de que ya tenemos npm y yarn) es instalar yarn. Para ello podemos ejecutar el siguiente comando en CMD:

Npm install --global yarn

```
Símbolo del sistema
Microsoft Windows [Versión 10.0.22621.1702]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\Patito>npm install --global yarn

> yarn@1.22.19 preinstall C:\Program Files\nodejs\node_modules\yarn
> ; (node ./preinstall.js > /dev/null 2>&1 || true)

C:\Program Files\nodejs\yarn -> C:\Program Files\nodejs\node_modules\yarn\bin\yarn.js
C:\Program Files\nodejs\yarnpkg -> C:\Program Files\nodejs\node_modules\yarn\bin\yarn.js
+ yarn@1.22.19
added 1 package in 1.154s

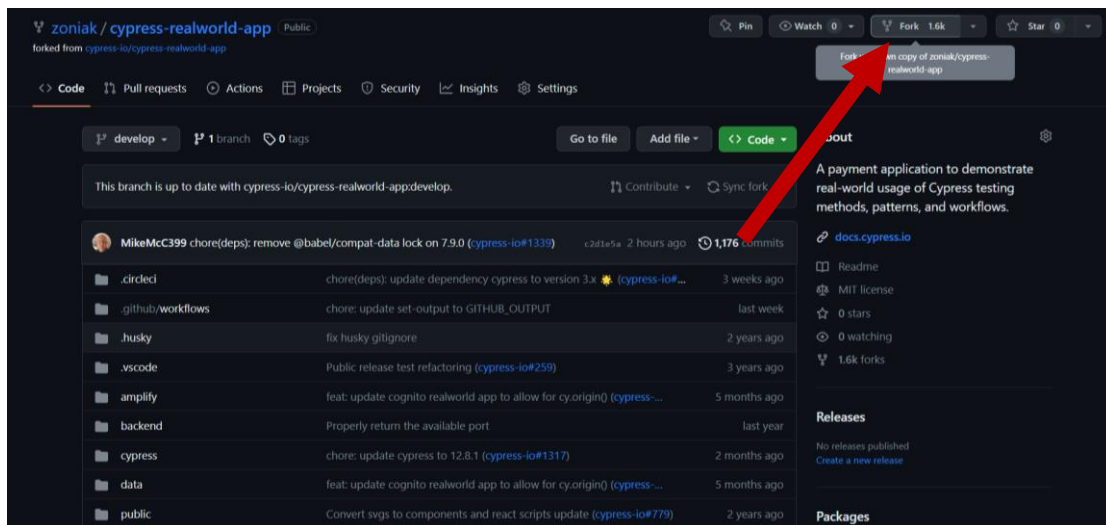
C:\Users\Patito>yarn -v
1.22.19

C:\Users\Patito>
```

Una vez instalado este ([si tenéis problemas con npm hay otras alternativas](#)) vamos a proceder a clonar el repositorio que nos ofrece Cypress con su aplicación real. Tenéis toda la documentación en Moodle en el apartado de real world test o en [este enlace](#).

2.2. FORK DE LA APLICACIÓN

¿Pero esperar!! No vamos simplemente a clonar el repositorio, vamos a hacer un fork!! Se que es demasiado increíble para ser cierto pero si vamos a usar ese comando con el que tanto os he dado la tabarra. Para clonar un repositorio en GitHub abrimos el repositorio y le damos aquí arriba:



El repositorio a clonar es el siguiente:

<https://github.com/cypress-io/cypress-realworld-app.git>

2.3. LANZAMIENTO DE LA APLICACIÓN

Una vez hecho el fork y clonado instalamos todas las dependencias necesarias en el directorio del proyecto con:

```
yarn install
```

Si nos da un error de este estilo:

```
PS C:\Users\Patito\Desktop\Trabajo\HLanz\Entornos de desarrollo\GitHub\cypress-realworld-app> yarn -v
yarn : No se puede cargar el archivo C:\Program Files\nodejs\yarn.ps1 porque la ejecución de scripts está deshabilitada en este
sistema. Para obtener más información, consulta el tema about_Execution_Policies en https://go.microsoft.com/fwlink/?LinkID=135170.
En línea: 1 Carácter: 1
+ yarn -v
```

Tenéis que cambiar vuestros permisos de ejecución de scripts con la PowerShell. [Aquí os dejo un artículo](#) de como hacerlo. Es muy fácil, simplemente abrimos el PowerShell modo admin y ejecutamos el siguiente comando (solo si te da error):

```
Set-ExecutionPolicy RemoteSigned -Scope CurrentUser
```

Después de instalar las dependencias necesarias ejecutamos la aplicación con:

```
yarn dev
```

La aplicación se debería ejecutar en <http://localhost:3000> . PEEEEERO, veréis esta aplicación usa el framework react, el cual no es compatible con versiones de node.js por encima de 15 (y seguramente tengáis una 20 o superior, según cuando leas esto). Para que te arranque la aplicación y no se quede eternamente arrancando tenéis que ejecutar lo siguiente en el terminal antes de ejecutar el yarn dev:

```
$env:NODE_OPTIONS="--openssl-legacy-provider"
```

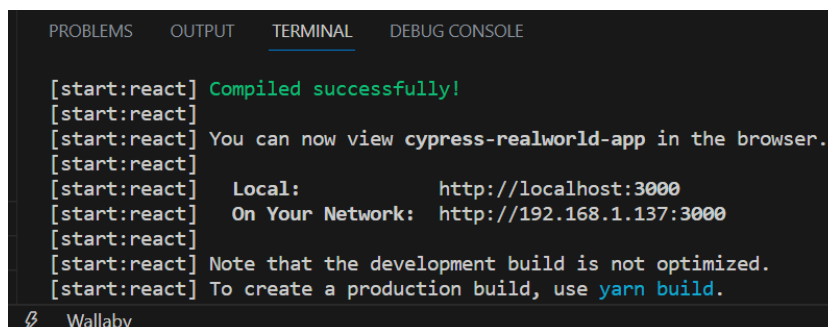
También podéis hacerlo con:

```
set NODE_OPTIONS=--openssl-legacy-provider
```

Si por algún casual tenéis Mac seria:

```
export NODE_OPTIONS=--openssl-legacy-provider
```

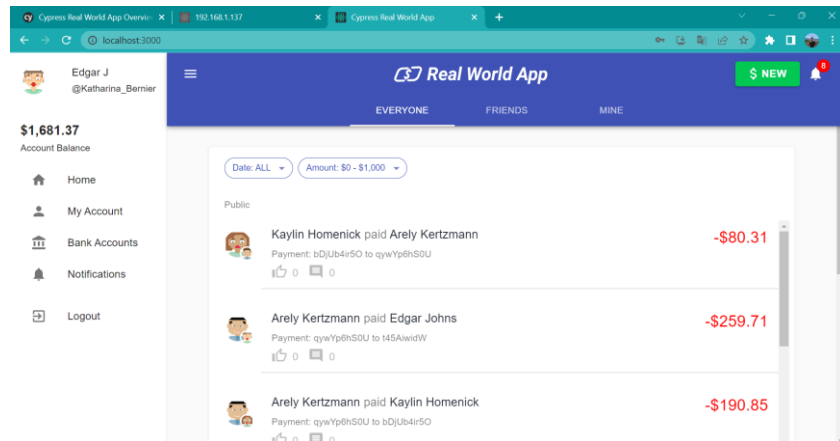
Darle las gracias de esto a mis alumnos Iyas y Rubén. Con el comando `yarn list:dev:users` podemos listar los usuarios de la aplicación, pero en sistemas Windows no os va a funcionar, por lo tanto usar el usuario **Katharina Bernier** y la contraseña es **s3cret**. Si no se te lanza en el navegador busca un mensaje de este estilo y prueba en el otro link:



```
PROBLEMS  OUTPUT  TERMINAL  DEBUG CONSOLE

[start:react] Compiled successfully!
[start:react]
[start:react] You can now view cypress-realworld-app in the browser.
[start:react]
[start:react]   Local:           http://localhost:3000
[start:react]   On Your Network: http://192.168.1.137:3000
[start:react]
[start:react] Note that the development build is not optimized.
[start:react] To create a production build, use yarn build.
```

Si arranca pero no os deja autentificaros con el usuario y la contraseña, probar a cerrar la terminal y volver a lanzar la app. Al abrirse y autenticarse se debería ver esto:



Ahora os toca meteros en la documentación oficial y probar a realizar algún test. Recordar que para abrir Cypress y realizar los test deberéis hacerlo con `yarn cypress:open`.