Contenido

[1. Autores del trabajo, planificación y entrega 3](#_Toc513225434)

[1.1 Autores 3](#_Toc513225435)

[1.2 Planificación 3](#_Toc513225436)

[1.3 Entrega 3](#_Toc513225437)

[2. Requisitos del prototipo a implementar 4](#_Toc513225438)

[2.1 Requisitos funcionales 4](#_Toc513225439)

[2.2 Otros requisitos 4](#_Toc513225440)

[3. Criterios de comparación en la implementación 6](#_Toc513225441)

[3.1 Criterio 1: Gráfico tarta 6](#_Toc513225442)

[3.2 Criterio 2: Gráfico de líneas 6](#_Toc513225443)

[3.3 Criterio 3: Gráfico múltiple eje x 6](#_Toc513225444)

[3.4 Criterio 4: Gráfico área stack 6](#_Toc513225445)

[3.5 Criterio 5: Gráfico de barras horizontales 6](#_Toc513225446)

[3.11 Criterio 11: Horas empleadas en el desarrollo 7](#_Toc513225447)

[3.12 Criterio 12: Velocidad de funcionamiento 7](#_Toc513225448)

[3.13 Criterio 13: Almacenamiento necesario para el desarrollo 7](#_Toc513225449)

[4. Proyecto de implementación de un prototipo del sistema utilizando la tecnología Chartjs 8](#_Toc513225450)

[4.1 Documentación de diseño 8](#_Toc513225451)

[4.2 Documentación de construcción 8](#_Toc513225452)

[4.3 Documentación de pruebas 10](#_Toc513225453)

[4.4 Documentación de instalación 10](#_Toc513225454)

[4.5 Manual de usuario 11](#_Toc513225455)

[5. Proyecto de implementación de un prototipo del sistema utilizando la tecnología Echarts 12](#_Toc513225456)

[5.1 Documentación de diseño 12](#_Toc513225457)

[5.2 Documentación de construcción 12](#_Toc513225458)

[5.3 Documentación de pruebas 16](#_Toc513225459)

[5.4 Documentación de instalación 17](#_Toc513225460)

[5.5 Manual de usuario 17](#_Toc513225461)

[6. Comparación de las dos implementaciones 18](#_Toc513225462)

[6.1 Evaluación de los criterios en la implementación usando la tecnología A 18](#_Toc513225463)

[6.2 Evaluación de los criterios en la implementación usando la tecnología B 18](#_Toc513225464)

[7. Comparación de la implementación de las tecnologías 19](#_Toc513225465)

[8. Conclusiones 21](#_Toc513225466)

# 1. Autores del trabajo, planificación y entrega

## 1.1 Autores

Componemos en grupo denominado como T4 y los integrantes son los siguientes:

* Urbano José Villanueva Rodríguez.
* Alberto Cabrera Plata.
* Fernando García Fernández.
* José Daniel Navarro Sierra.
* Martina Andrea Palomino Berrocal.

## 1.2 Planificación

En este apartado se debe incluir un enlace (URL) compartido a la planificación del trabajo utilizando una herramienta online de diagramación Gantt (por ejemplo, GanttPro, versión gratuita).

Hay que tener en cuenta que cada participante del grupo debe tener asignadas tareas que sumen al menos 45 horas. El peso de este trabajo en la calificación total de la asignatura es de un 30%, por tanto requiere de una dedicación de 45 horas del total de 150 horas de la asignatura.

## 1.3 Entrega

Enlace a GitHub: <https://github.com/UrbanoJVR/TG3>

* Informe del trabajo: con el nombre TG3\_final.docx
* Presentación del trabajo: TG3\_final.pptx
* Prototipos obtenidos implementando cada una de las tecnologías (deben incluir el código fuente y todos los archivos necesarios para la instalación y uso de cada prototipo):
  + PrototipoTecnologiaA\_final.zip (o .rar)
  + PrototipoTecnologiaB\_final.zip (o .rar).

Dichos archivos serán los que se tendrán en cuenta para la calificación del trabajo.

# 2. Requisitos del prototipo a implementar

El objetivo del proyecto es comparar la implementación de mismos prototipos utilizando las tecnologías Chartjs y Echarts.

En primer lugar, comparamos algunos de los gráficos más significativos que existen en ambas librerías, que son los siguientes: gráficos tarta, gráfico de barras, gráfico de líneas, gráficos múltiples, gráfico de ejes-x, gráficos de barras horizontales y gráficos radar.

En segundo lugar, realizamos la implementación de gráficos de árbol y mapa de densidad, que solo existen para la segunda librería estudiada, Echarts.

Finalmente, y como punto fuerte de la comparación entre ambas librerías, mostramos la implementación del gráfico de líneas de cada una de ellas a fin de mostrar datos de una API REST sobre el precio del bitcoin en USD, EUR y GBP.

## 2.1 Requisitos funcionales

| **REQ.** | **DESCRIPCIÓN** |
| --- | --- |
| RF01 | La aplicación permite mostrar el gráfico tarta en ambas tecnologías |
| RF02 | La aplicación permite mostrar el gráfico de barras en ambas tecnologías |
| RF03 | La aplicación permite mostrar el gráfico de líneas en ambas tecnologías |
| RF04 | La aplicación permite mostrar el gráfico múltiple en ambas tecnologías |
| RF05 | La aplicación permite mostrar el gráfico de ejes-x en ambas tecnologías |
| RF06 | La aplicación permite mostrar el gráfico de barras horizontales |
| RF07 | La aplicación permite mostrar el gráfico de tipo radar |
| RF08 | La aplicación permite mostrar un gráfico de líneas a fin de mostrar datos de una API REST sobre el precio de bitcoin en USD, EUR y GBP |
| RF09 | Mostrar leyenda de los gráficos |
| RF10 | Activar los datos que se muestran en los gráficos |
| RF11 | Desactivar los datos que se muestran en los gráficos |
| RF12 | Guardar gráfico en formato imagen |

## 2.2 Otros requisitos

USA: Usabilidad

MAN: Mantenibilidad

TR: Tiempo de respuesta

SEG: Seguridad

LAN: Lenguajes de alto nivel

REU: Reusabilidad

| **REQ.** | **DESCRIPCIÓN** |
| --- | --- |
| USA01 | La aplicación debe facilitar la navegación a través de sus páginas. |
| USA02 | La aplicación debe ser perfectamente manejable por cualquier tipo de usuario. |
| MAN01 | Cuidado en las fases de diseño, codificación y prueba. |
| MAN02 | Configuración adecuada del producto software. |
| MAN03 | Adecuada cualificación del equipo de desarrolladores del software. |
| MAN04 | Estructura del software fácil de comprender. |
| MAN05 | Estructura estandarizada de la documentación. |
| MAN06 | Documentación disponible de los casos de prueba. |
| TR01 | Todas las respuestas del sistema deben producirse en un tiempo máximo de 30 segundos. |
| SEG01 | La aplicación no perderá de forma accidental los datos. |
| SEG02 | El almacenamiento de información sensible, tanto propia de la lógica de la aplicación como las credenciales de acceso, debe de almacenarse cifrada en todos los servidores, y especialmente en el de base de datos. |
| LAN01 | Empleo de lenguajes de programación estandarizados. |
| LAN02 | Empleo de Javascript |
| REU01 | La aplicación reutilizará clases comunes. |

# 3. Criterios de comparación en la implementación

## 3.1 Criterio 1: Gráfico tarta

*Descripción: posibilidad de creación de un gráfico tarta.*

*Tipo de valor: booleano.*

## 3.2 Criterio 2: Gráfico de líneas

*Descripción: posibilidad de creación de un gráfico de líneas.*

*Tipo de valor: booleano.*

## 3.3 Criterio 3: Gráfico múltiple eje x

*Descripción: posibilidad de creación de un gráfico múltiple.*

*Tipo de valor: booleano.*

## 3.4 Criterio 4: Gráfico área stack

*Descripción: posibilidad de creación de un gráfico de eje-x.*

*Tipo de valor: booleano.*

## 3.5 Criterio 5: Gráfico de barras horizontales

*Descripción: posibilidad de creación de un gráfico de barras horizontales.*

*Tipo de valor: booleano.*

**3.6 Criterio 6: Gráfico de tipo radar**

*Descripción: posibilidad de creación de un gráfico de tipo radar.*

*Tipo de valor: booleano.*

**3.7 Criterio 7: Gráfico de mapa de densidad**

*Descripción: posibilidad de creación de un gráfico de mapa de densidad.*

*Tipo de valor: booleano.*

**3.8 Criterio 8: Gráfico de árbol**

*Descripción: posibilidad de creación de un gráfico de árbol.*

*Tipo de valor: booleano.*

**3.9 Criterio 9: Gráfico de líneas de datos del API REST de bitcoin**

*Descripción: posibilidad de creación de un gráfico tarta.*

*Tipo de valor: booleano.*

**3.10 Criterio 10: Líneas de código**

*Descripción: cantidad de líneas de código que han sido necesarias para la implementación de los gráficos.*

*Tipo de valor: numérico.*

## 3.11 Criterio 11: Horas empleadas en el desarrollo

*Descripción: horas necesarias que se han empleado en el desarrollo de los gráficos.*

*Tipo de valor: numérico.*

## 3.12 Criterio 12: Velocidad de funcionamiento

*Descripción: velocidad de funcionamiento de los gráficos.*

*Tipo de valor: numérico.*

## 3.13 Criterio 13: Almacenamiento necesario para el desarrollo

*Descripción: cantidad de memoria necesaria del ordenador para la realización y funcionamiento de los gráficos.*

*Tipo de valor: numérico.*

# 4. Proyecto de implementación de un prototipo del sistema utilizando la tecnología Chartjs

## 4.1 Documentación de diseño

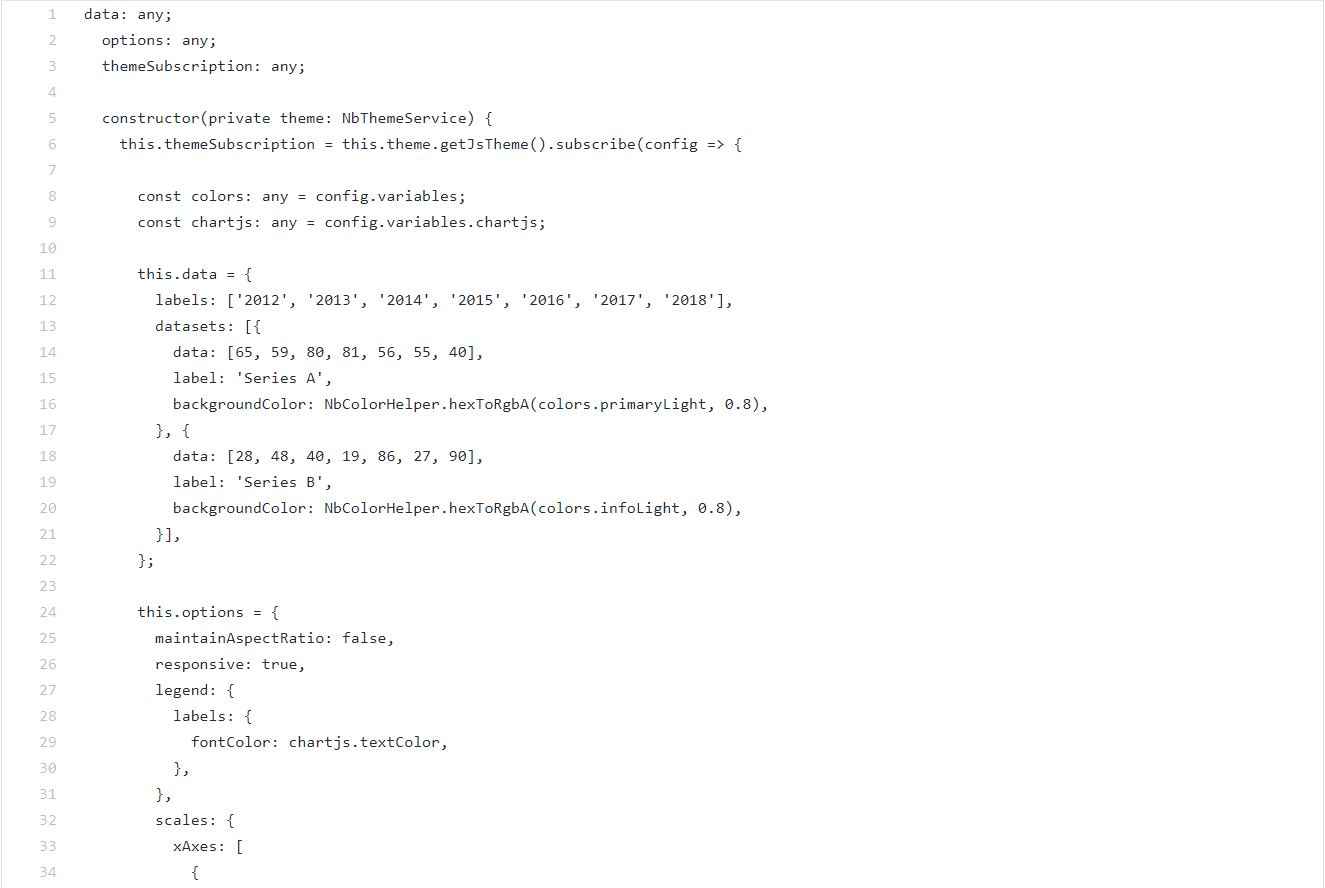
Hay que incluir la descripción del diseño del prototipo, incluyendo diagramas, y el diseño de la interfaz de usuario.

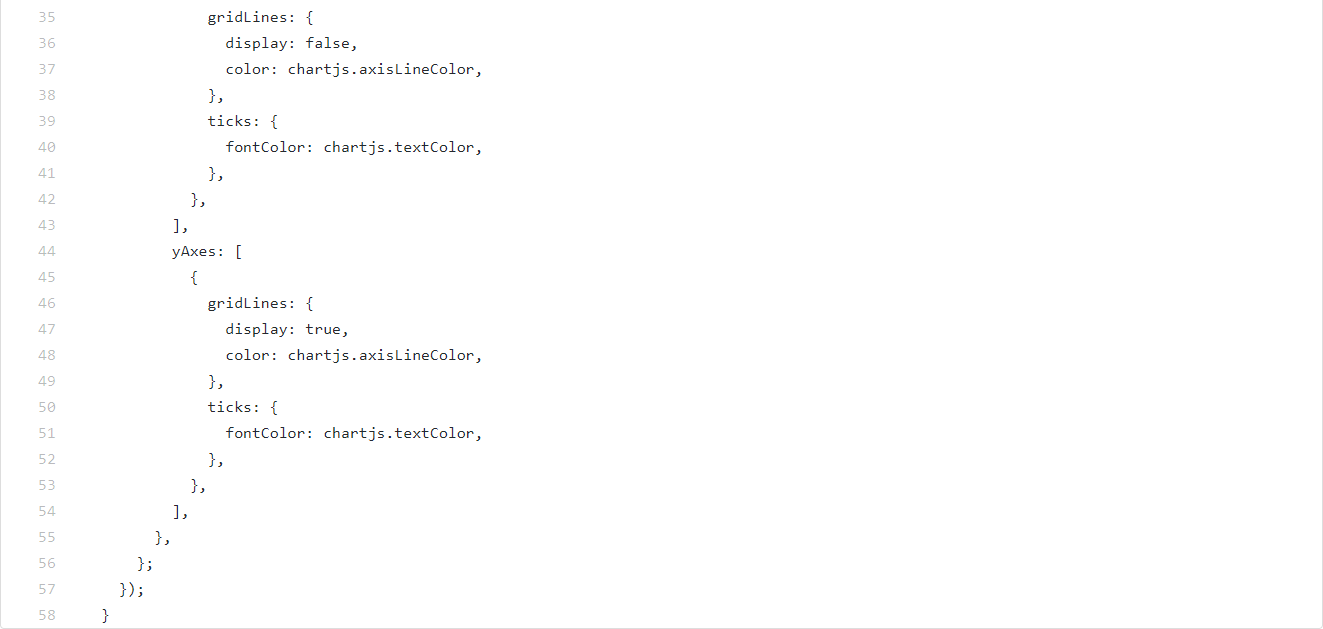
\*Pantallazo menú principal\*

\*pantallazo gráficos\*

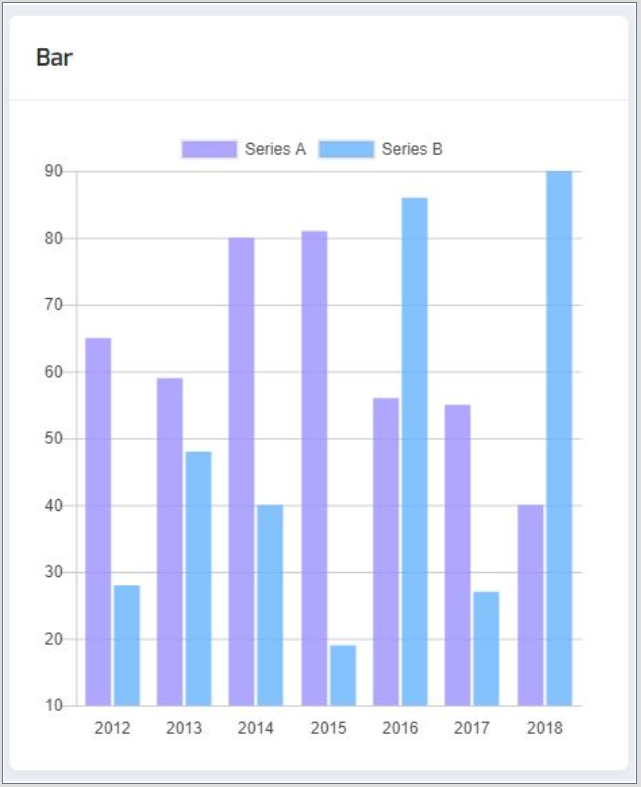
## 4.2 Documentación de construcción

A continuación, mostramos una captura de pantalla del código referente a la construcción del gráfico que hemos elegido para mostrar de Chartjs, que es el gráfico de barras:





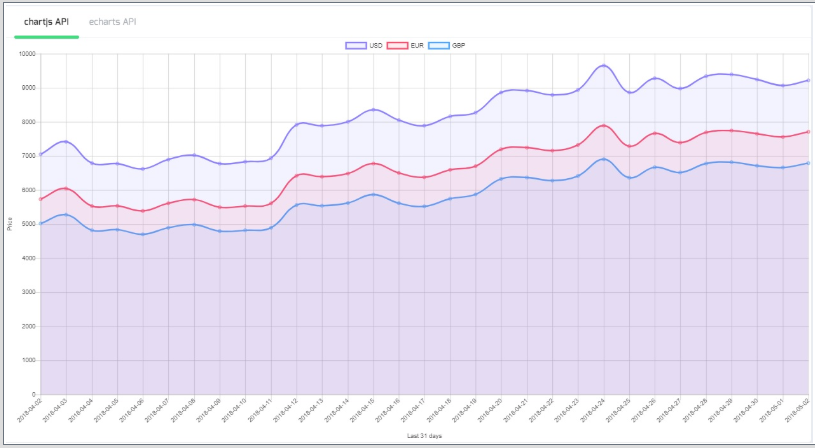
Y como resultado obtenemos:



Por otro lado, mostramos el código correspondiente a la realización del gráfico de líneas modo “live” que lee de una API REST los valores del Bitcoin diarios y los muestra en USD, EUR y GBP:



Y como resultado obtenemos:



## 4.3 Documentación de pruebas

Casos de prueba establecidos y resultados de las pruebas y acciones de corrección. No es creíble que no hayan aparecido errores en los caso de prueba.

## 4.4 Documentación de instalación

Describiremos paso a paso el proceso para instalar el prototipo:

1. Instalar en la computadora Visual Studio Code como IDE.
2. Descargar Node.js.
3. Descargar la carpeta “charts-comparison” de <https://github.com/UrbanoJVR/TG3>.
4. Una vez descargada la carpeta anterior, abrirla con VS Code.
5. Ejecutar CTRL + Ñ para abrir la terminal en el IDE.
6. Ejecutar en la terminal el comando “npm install @angular/cli”.
7. Ejecutar en la terminal el comando “npm install” para instalar las dependencias node que necesita el proyecto.
8. Ejecutar en la termina el comando “ng serve -o” para que la aplicación corra en localhost y se abra directamente la web en el navegador por defecto.

## 4.5 Manual de usuario

Una vez realizados los pasos en el apartado 4.4, y la aplicación se muestre en el navegador que tengamos establecido por defecto en nuestra computadora, observaremos que la aplicación consta de un menú en la parte izquierda dispuesto con tres pestañas:

* Chartjs: nos redirige a los gráficos de Chartjs.
* Echarts: nos redirige a los gráficos de Echarts.
* Live: nos redirige a los gráficos de líneas utilizados para la representación de los datos del API REST sobre bitcoin.

Tanto con Chartjs como con Echarts hemos implementado los siguientes gráficos: gráficos tarta, gráfico de barras, gráfico de líneas, gráficos múltiples, gráfico de ejes-x, gráficos de barras horizontales y gráficos radar.

Adicionalmente Echarts, presenta dos tipos de gráficos más: gráficos de árbol y mapa de densidad; ya que Chartjs no los dispone.

Dentro de cada gráfico existe un botón referente a la leyenda del gráfico en cuestión, con el fin de activar o desactivar los datos que se desean mostrar.

Alguno de los gráficos, además de la leyenda posee un icono para guardar el gráfico en formato imagen.

# 5. Proyecto de implementación de un prototipo del sistema utilizando la tecnología Echarts

## 5.1 Documentación de diseño

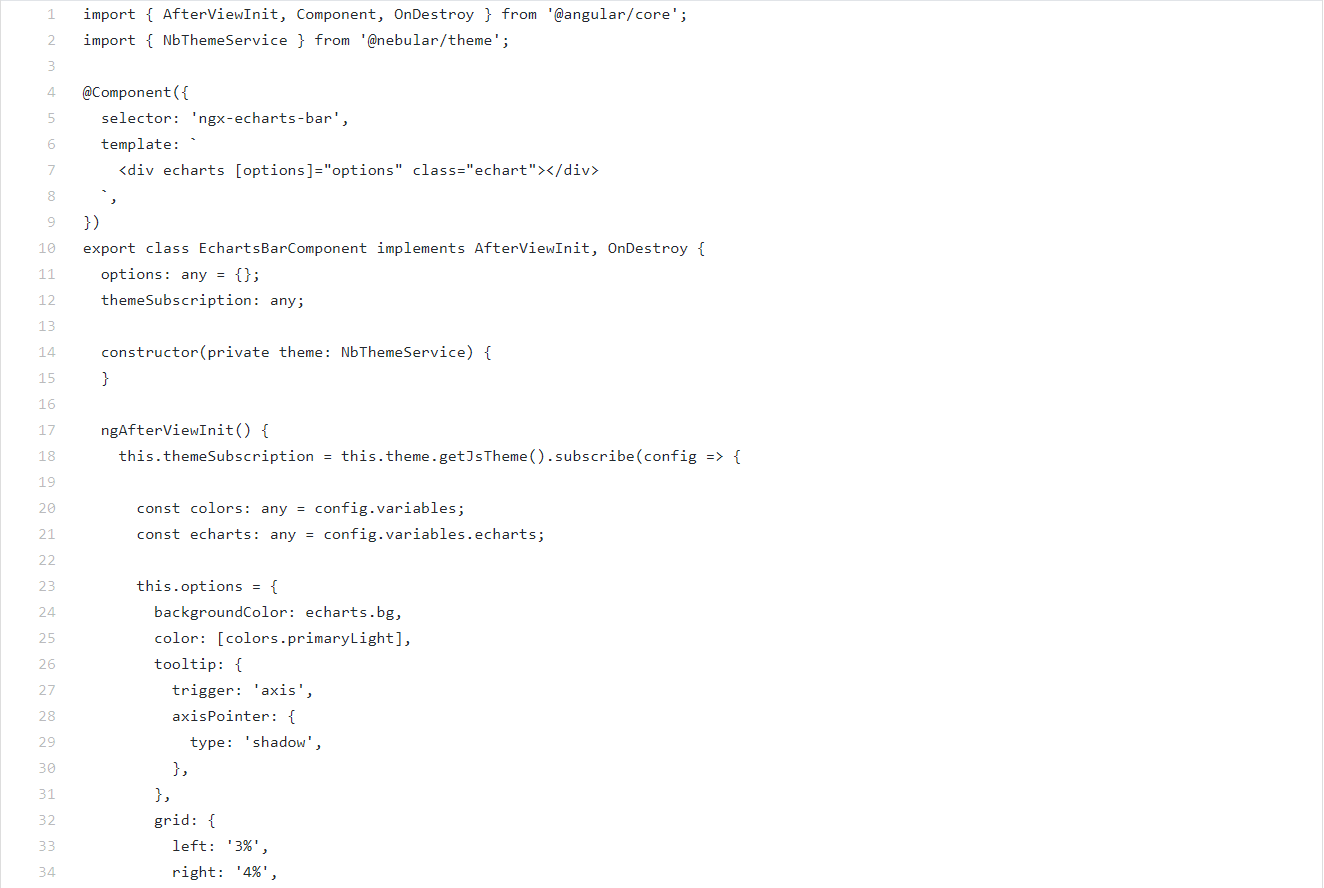
Hay que incluir la descripción del diseño del prototipo, incluyendo diagramas, y el diseño de la interfaz de usuario.

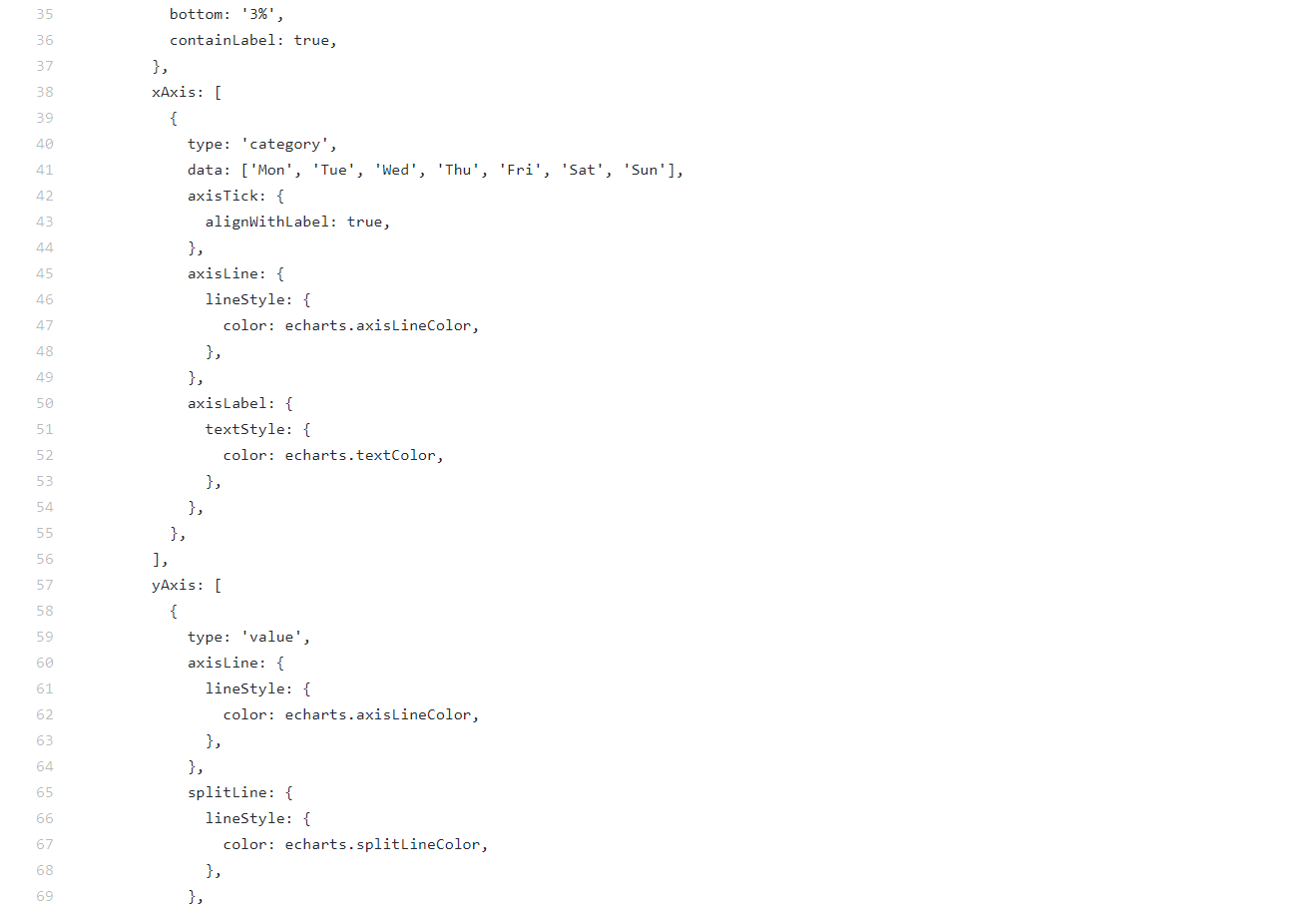
\*Pantallazo menú principal\*

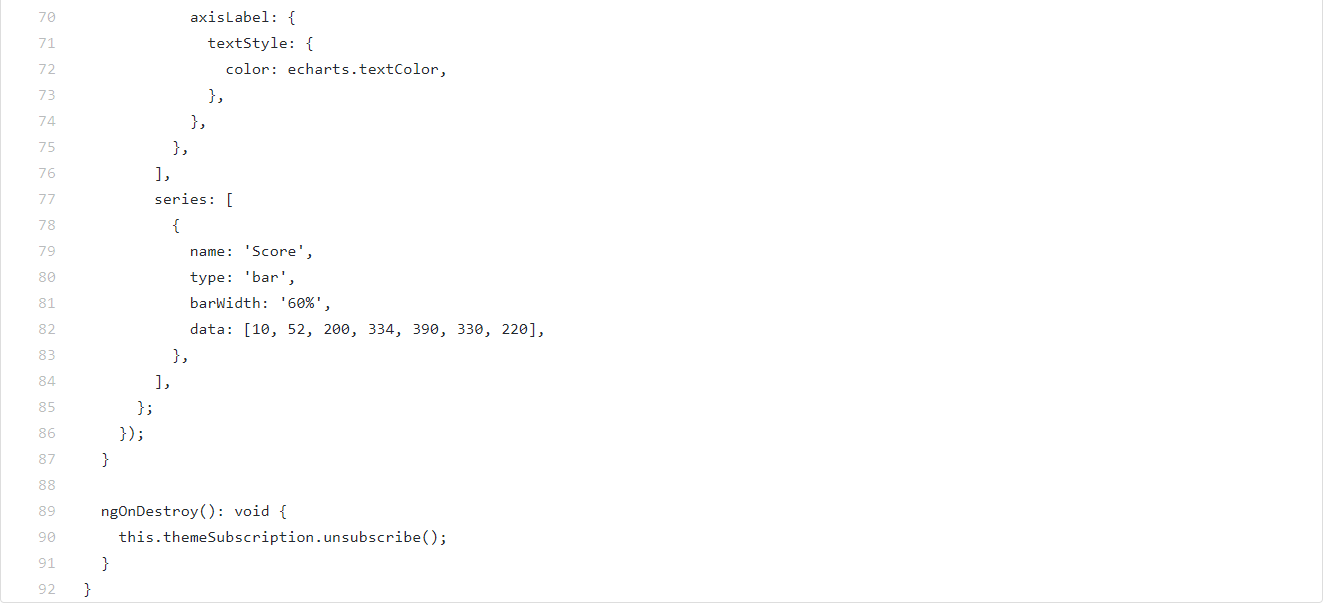
\*pantallazo gráficos\*

## 5.2 Documentación de construcción

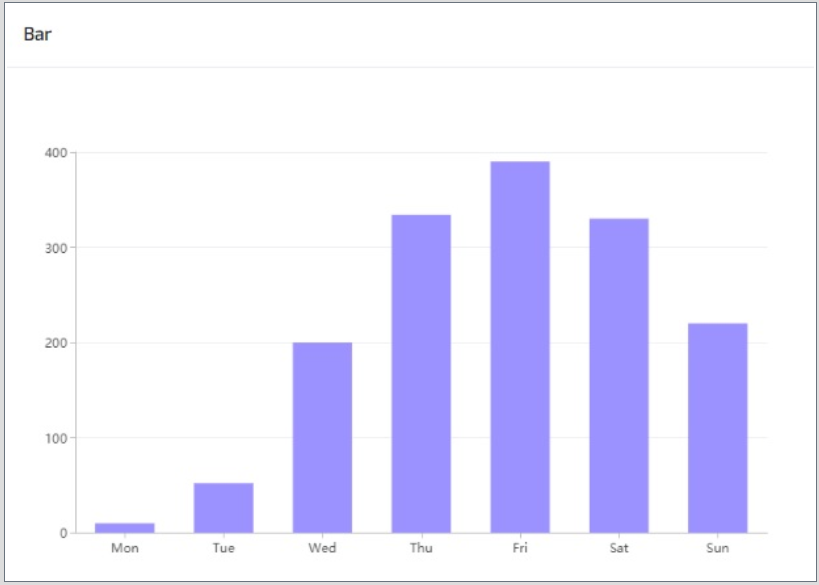
A continuación, mostramos una captura de pantalla del código referente a la construcción del gráfico que hemos elegido para mostrar de Echarts, que es el gráfico de barras:



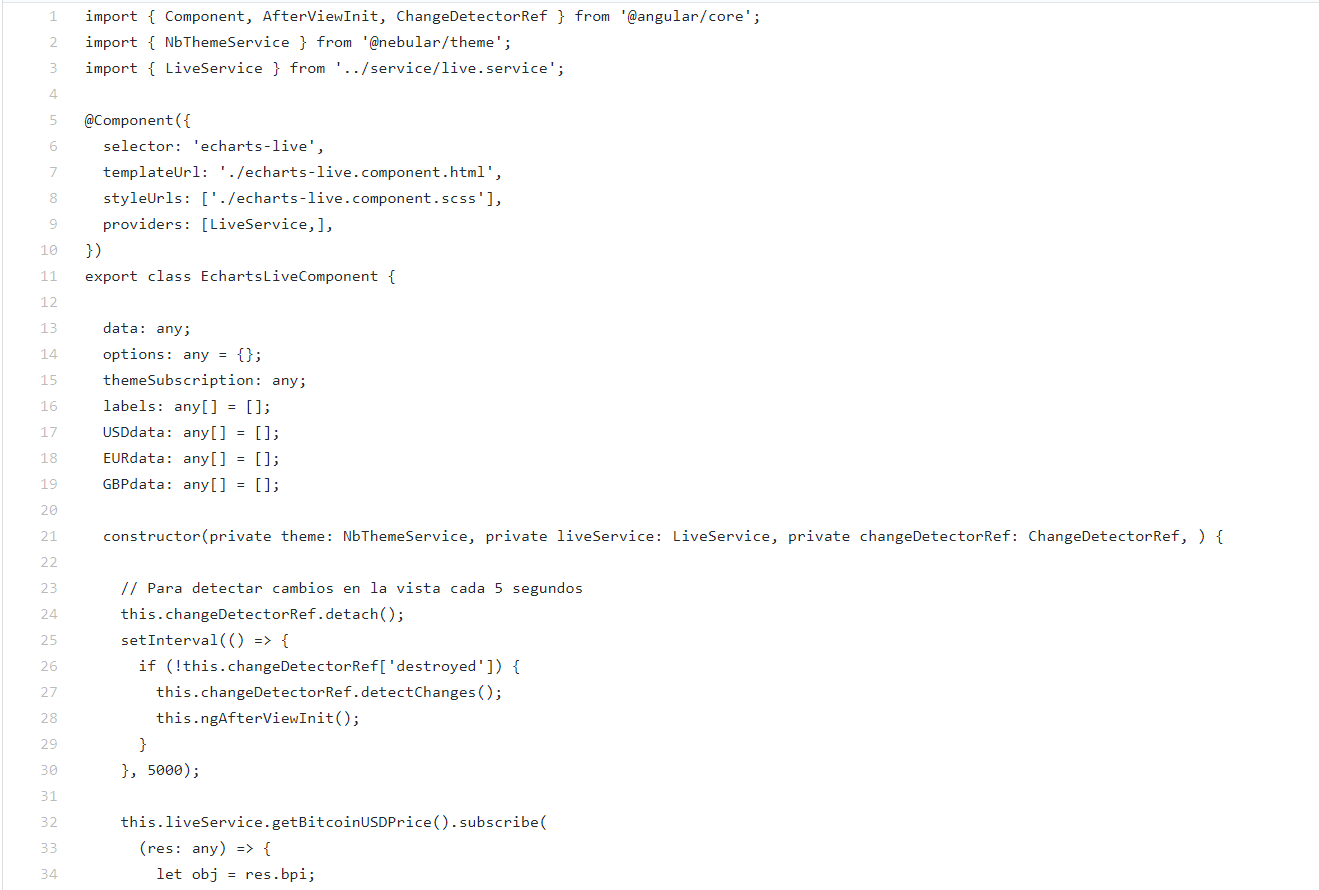


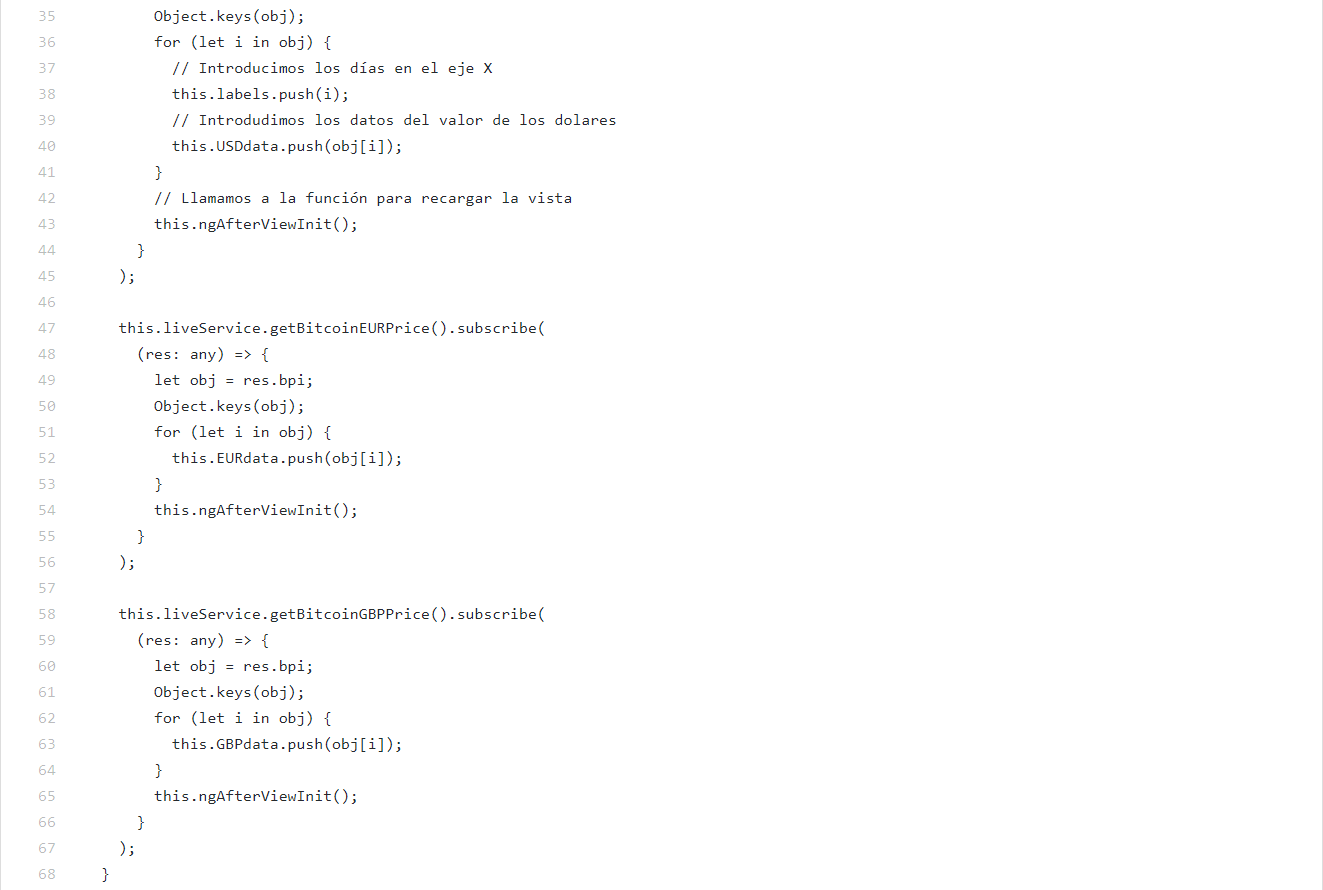
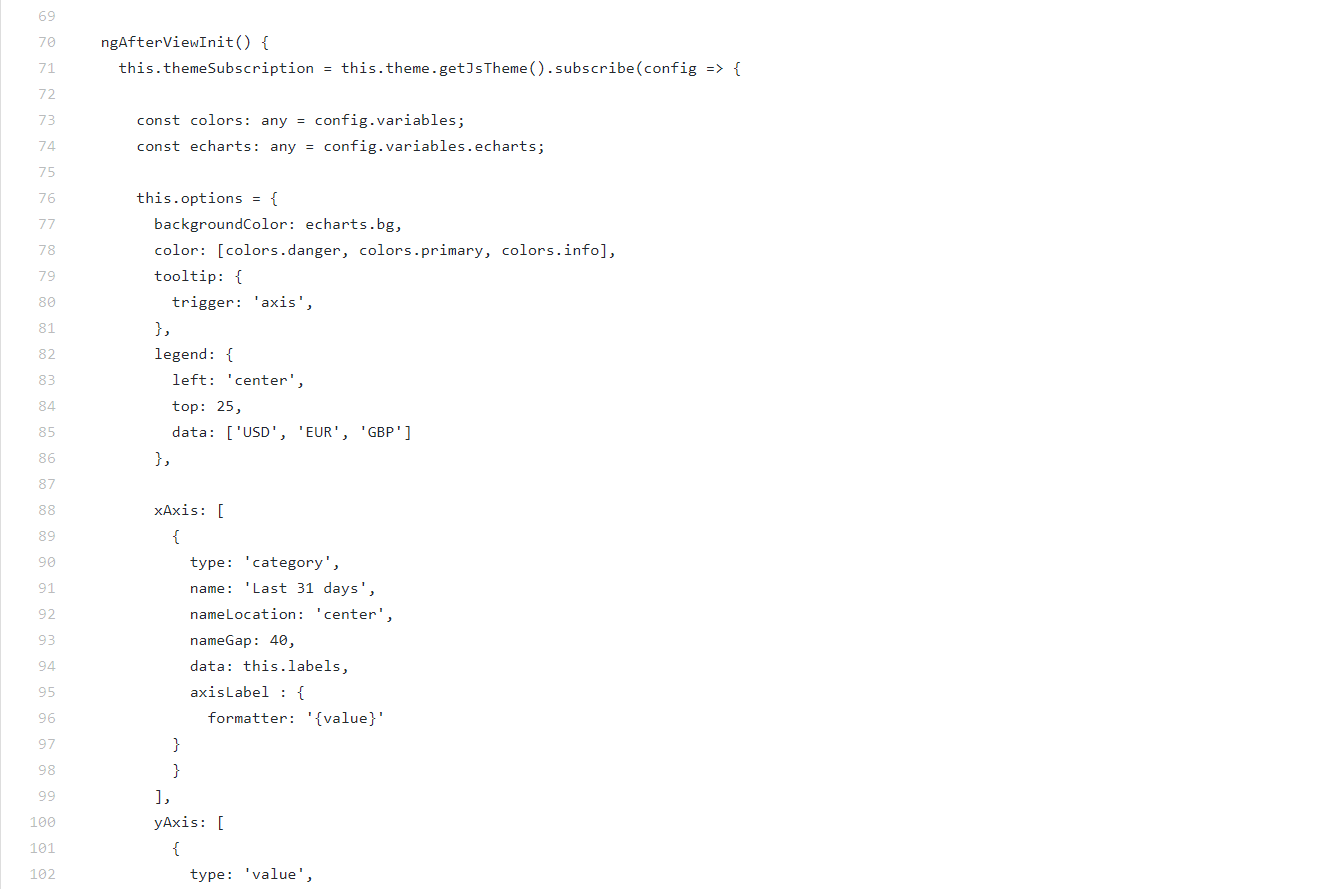
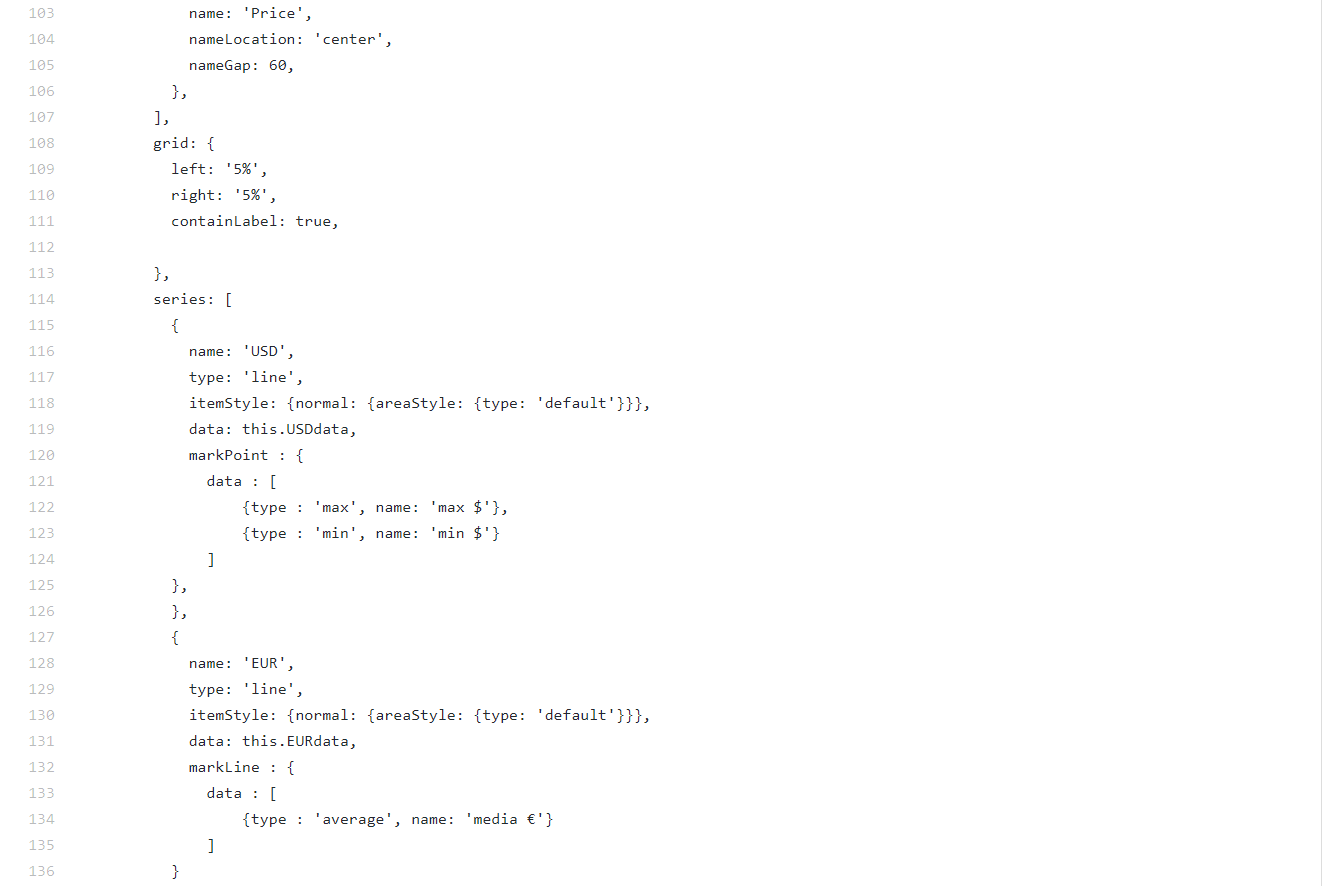
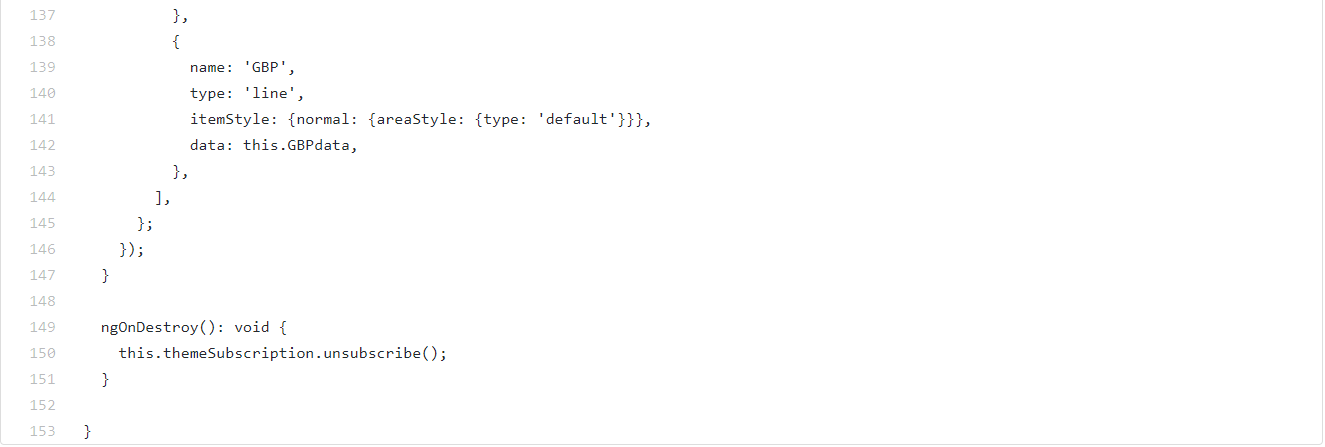


Y como resultado obtenemos:

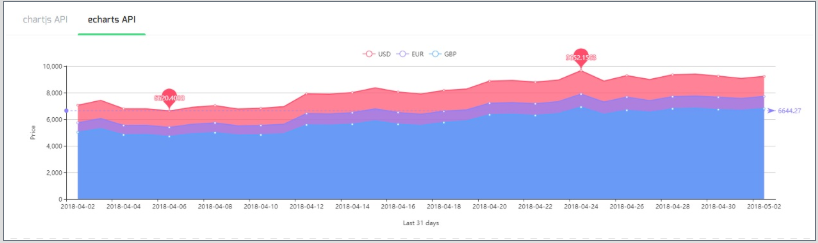


Por otro lado, mostramos el código correspondiente a la realización del gráfico de líneas modo “live” que lee de una API REST los valores del Bitcoin diarios y los muestra en USD, EUR y GBP:



Y como resultado obtenemos:



## 5.3 Documentación de pruebas

Casos de prueba establecidos y resultados de las pruebas y acciones de corrección. No es creíble que no hayan aparecido errores en los caso de prueba.

## 5.4 Documentación de instalación

La documentación de instalación del prototipo para Echarts es la misma que para Chartjs ya que se presentan conjuntamente en la misma aplicación:

Describiremos paso a paso el proceso para instalar el prototipo:

1. Instalar en la computadora Visual Studio Code como IDE.
2. Descargar Node.js.
3. Descargar la carpeta “charts-comparison” de <https://github.com/UrbanoJVR/TG3>.
4. Una vez descargada la carpeta anterior, abrirla con VS Code.
5. Ejecutar CTRL + Ñ para abrir la terminal en el IDE.
6. Ejecutar en la terminal el comando “npm install @angular/cli”.
7. Ejecutar en la terminal el comando “npm install” para instalar las dependencias node que necesita el proyecto.
8. Ejecutar en la termina el comando “ng serve -o” para que la aplicación corra en localhost y se abra directamente la web en el navegador por defecto.

## 5.5 Manual de usuario

El manual para el usuario es el mismo para Chartjs que para Echarts ya que ambas tecnologías se han utilizado para la creación de los mismos gráficos, y adicionalmente dos más de Echarts, en una misma aplicación.

Una vez realizados los pasos en el apartado 4.4, y la aplicación se muestre en el navegador que tengamos establecido por defecto en nuestra computadora, observaremos que la aplicación consta de un menú en la parte izquierda dispuesto con tres pestañas:

* Chartjs: nos redirige a los gráficos de Chartjs.
* Echarts: nos redirige a los gráficos de Echarts.
* Live: nos redirige a los gráficos de líneas utilizados para la representación de los datos del API REST sobre bitcoin.

Tanto con Chartjs como con Echarts hemos implementado los siguientes gráficos: gráficos tarta, gráfico de barras, gráfico de líneas, gráficos múltiples, gráfico de ejes-x, gráficos de barras horizontales y gráficos radar.

Adicionalmente Echarts, presenta dos tipos de gráficos más: gráficos de árbol y mapa de densidad; ya que Chartjs no los dispone.

Dentro de cada gráfico existe un botón referente a la leyenda del gráfico en cuestión, con el fin de activar o desactivar los datos que se desean mostrar.

Alguno de los gráficos, además de la leyenda posee un icono para guardar el gráfico en formato imagen.

# 6. Comparación de las dos implementaciones

Se trata de dar valores a los criterios de comparación definidos en el apartado 3 sobre la implementación de cada uno de los prototipos.

## 6.1 Evaluación de los criterios en la implementación usando la tecnología A

Debe incluir al menos una tabla con la siguiente estructura.

| **CRITERIO** | **EVALUACIÓN** |
| --- | --- |
| Criterio 1 |  |
| Criterio 2 |  |
| … |  |
| Criterio N |  |

Y algunos comentarios aclaratorios sobre aquellos criterios cuyo valor indicado en la tabla no sea suficiente para entenderlo.

## 6.2 Evaluación de los criterios en la implementación usando la tecnología B

# 7. Comparación de la implementación de las tecnologías

Debe incluir al menos una tabla resumen, en sección de página horizontal, cruzando los criterios y los valores de cada tecnología. Con una columna de comentarios sobre la comparación

| **CRITERIOS** | **TECNOLOGÍA A** | **TECNOLOGÍA B** | **COMENTARIOS** |
| --- | --- | --- | --- |
| 1 |  |  |  |
| 2 |  |  |  |
| … |  |  |  |
| N |  |  |  |

# 8. Conclusiones

A partir de la información incluida en el apartado 7 y de la experiencia al realizar el trabajo, el grupo debe estar en condiciones de manifestar su opinión sobre la implementación del sistema utilizando ambas tecnologías, y debe plasmarla en este apartado, indicando las ventajas e inconvenientes más relevantes de utilizar una u otra tecnología para implementar el sistema.

---------------------------

(Hay que cumplir la estructura básica indicada de secciones. Pero si se desea se pueden añadir otras secciones como anexos. Por ejemplo, alguna encuesta de opinión realizada sobre las tecnologías, etc.)