# Análisis de hábitos de consumo en tarjetas de crédito usando Api BBVA.

Alumnos:

* Carlos Alcalá-Zamora Acero
* Dan Gonzalvo Guillén

# Planteamiento del problema.

BBVA PayStats recoge datos anónimos y estadísticos de millones de transacciones realizadas en las terminales POS de BBVA. Ofrece información de forma diaria desde 2014, disponible con datos geográficos o con código postal, podemos consultar datos básicos estadísticos de la factura de cada cliente, el número de transacciones o tarjetas usadas.

Con esta información podemos ser capaces de saber el tipo de consumidores que hay en una zona o sus patrones de consumo y en general estadísticas que puedan ayudar a la toma de decisiones para los negocios de la zona, como, por ejemplo, que tipos de productos y servicios se deben ofrecer o el horario que debería tener un comercio para optimizar sus ventas.

# Objetivo general y objetivos específicos, tecnológicos y de negocio.

Nuestro objetivo principal es la recopilación de la información para su tratamiento, obteniendo las métricas principales que nos permitan entender el comportamiento de los clientes. Una vez finalizado el proyecto podremos responder de forma clara y de un solo vistazo a cuestiones como:

* Tipos de compras realizadas.
* Origen de los clientes.
* Gasto de los clientes.
* Localización y tipo de gasto.

Para ello comenzaremos recopilando la información en formato JSON ofrecida por el api, al ser información ya recogida no necesitaremos un método de almacenamiento previo, por lo que enlazaremos el api con el tratamiento back en python. Recopilaremos toda la información y la filtraremos para generar conjuntos de datos que podamos representar y nos permitan contestar a preguntas como las anteriores. Una vez tratada la información ofreceremos una web sencilla en la que integraremos las gráficas y datos. Esta página web la publicaremos en una plataforma como Firebase para que sea accesible desde cualquier navegador web. De esta forma cualquier persona podrá visualizar las estadísticas de gasto de los usuarios, constituyendo una información muy valiosa para cualquier emprendedor que quiera abrir su negocio.

## Aportaciones fundamentales esperadas.

Nuestro objetivo es ofrecer una web clara y sencilla que nos muestre un conjunto de gráficas que interpreten los datos ofrecidos por el api. Esta web será accesible y nos mostrará los resultados para poder responder a las preguntas planteadas.

# Breve estado del arte.

Actualmente se disponen de aplicaciones privadas de pago que ofrecen una información similar para cada una de las entidades bancarias, pero de forma pública no se dispone de esta información. Gracias al api de BBVA podemos acceder a estos datos, y con este pequeño trabajo podremos ofrecer toda esta información a los nuevos emprendedores.

De esta manera el banco cumple con la directiva de servicios de pagos (PSD2) de la Unión Europea en las que las entidades financieras de la eurozona deben proporcionar acceso a la información de sus clientes a terceros cuando el cliente haya dado su consentimiento explícito. La fecha límite que ha dado la Unión Europea para llevar a cabo esta directiva es enero de 2018. Esta normativa pretende promover un mercado con mayor igualdad de oportunidades promoviendo la entrada de nuevos proveedores y servicios a la vez que protege de manera más fiable a los clientes disminuyendo el fraude.

# Planteamiento de la propuesta.

## Metodología a seguir.

Con una metodología Ágil, en un equipo de dos trabajadores y con la ayuda de un director de proyecto, hemos afrontado el reto. Para la gestión del proyecto hemos utilizado la herramienta Taiga de gestión de proyectos. En ella se ha creado el proyecto para la gestión del equipo y organización del desarrollo del proyecto. Taiga nos permite recoger todos los cambios realizados. Podemos visualizar todas las tareas directamente desde la interfaz de Taiga: <https://tree.taiga.io/project/danciff-apibbva/>

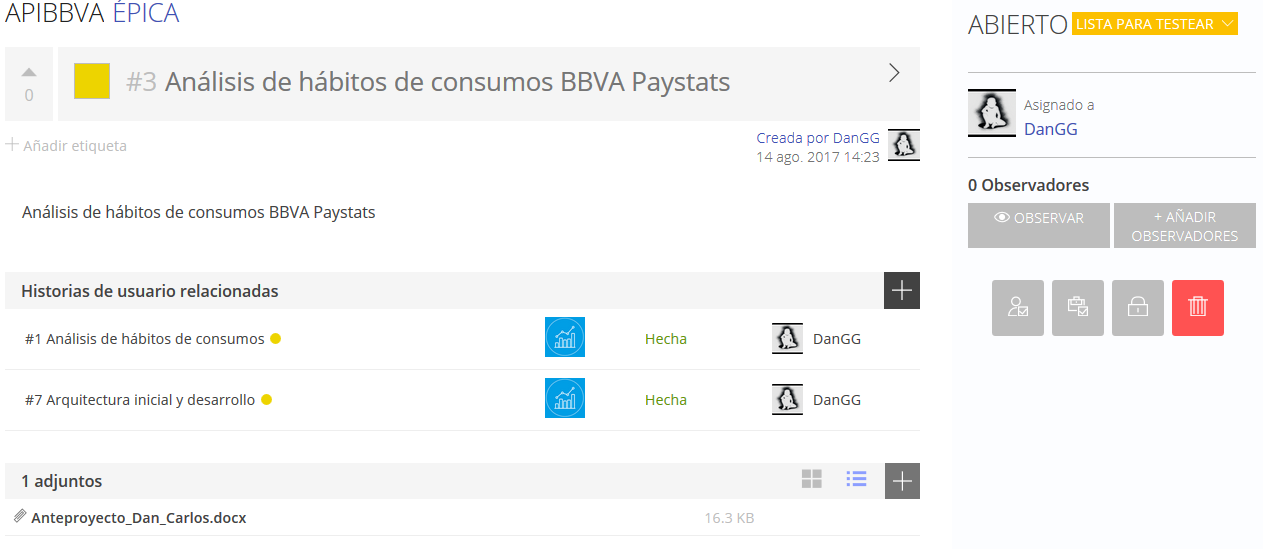


El proyecto se ha organizado en 4 tareas épicas, las cuales engloban todo el proyecto:



1. La primera de las tareas épicas es Análisis de hábitos de consumos BBVA Paystats, esta dispone de dos historias relacionadas:

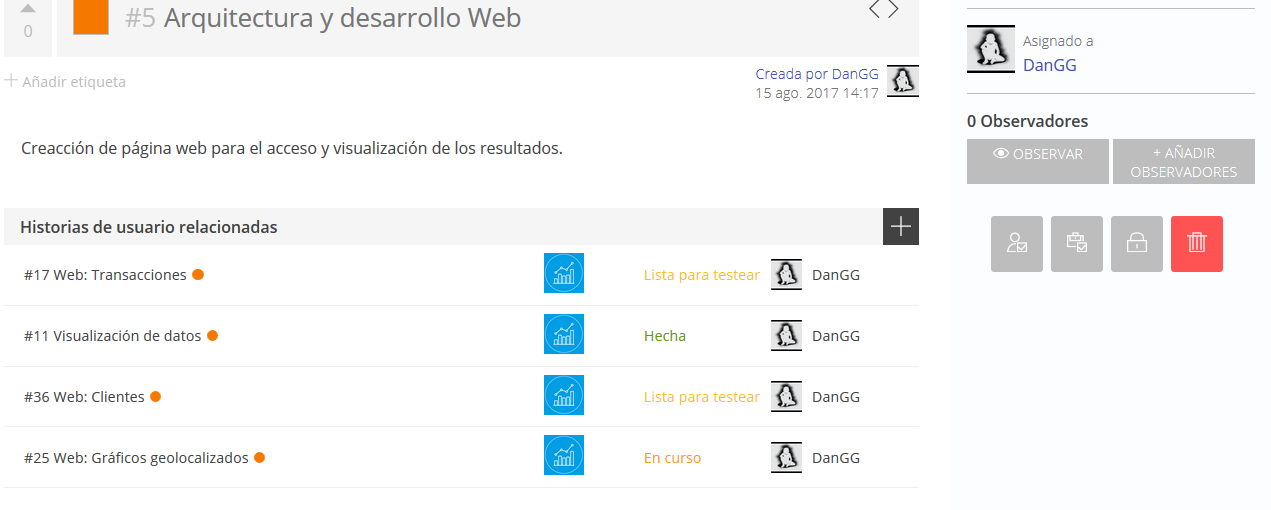
* Análisis de hábitos de consumos: Estudio de los datos ofrecidos y funcionamiento de Paystats.
* Arquitectura inicial y desarrollo: Diseño de la arquitectura inicial para cumplir los objetivos principales del proyecto.



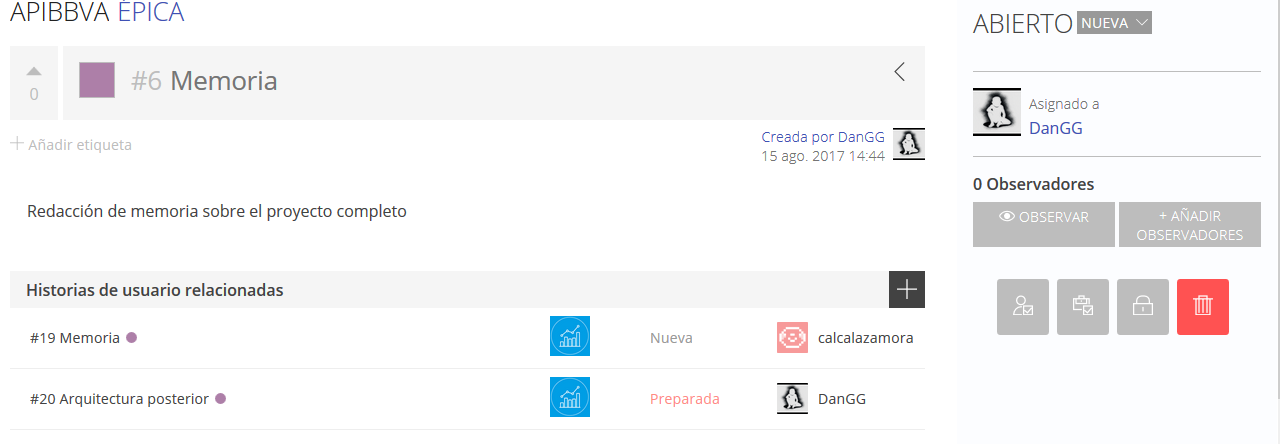
1. La segunda de las tareas se centra en el análisis y desarrollo de datos para la creación de gráficos, a su vez engloba un conjunto de tareas que referencian a los diferentes tipos de datos que podemos obtener:
   * Procesamiento de datos: Estudio general.
   * Localización: Relativo al estudio de los datos según el origen.
   * Categoría: Filtrado de datos de transacciones según el tipo.
   * Clientes: Relacionado con el análisis de las transacciones según el tipo de cliente.



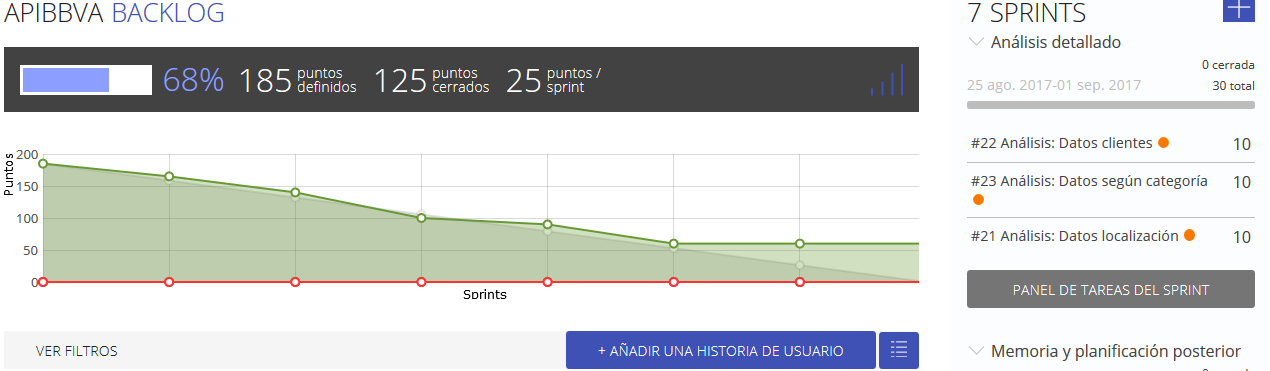
1. La tercera tarea está relacionada con la visualización de resultados, siguiendo el patrón de análisis, contiene 4 tareas:
   * Visualización: Corresponde a la estructura principal de la web.
   * Transacciones: Nos muestra el conjunto de gráficas relacionadas con los datos de transacciones analizados.
   * Clientes: Visualización de datos según el tipo de cliente.
   * Localización: Datos según el origen o destino de las transacciones.



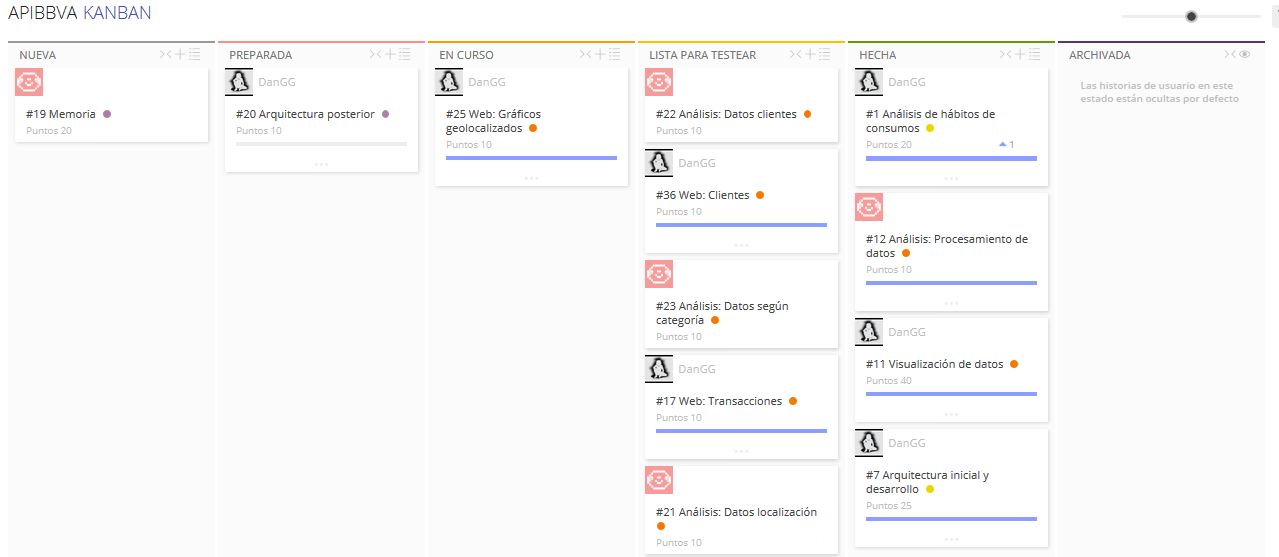
1. Por último tenemos la tarea de Memoria, que engloba la documentación del proyecto y su planificación futura:
   * Memoria: Generación de toda la documentación necesaria.
   * Arquitectura: Diseño del posible desarrollo para adaptar el proyecto para el funcionamiento con datos actualizados de forma continua.



El backlog del proyecto nos permite añadir y organizar las tareas, todas las tareas se han organizado en 7 Sprints, el inicio definitivo se ha realizado el 24 de Julio de 2017, y el proyecto finaliza el 4 de Septiembre de 2017.



Por último disponemos de una vista Kanban que nos permite de un solo vistazo comprobar el estado de las tareas y en qué fase de desarrollo se encuentran.



Todo lo anterior nos permite realizar una gestión clara y ordenada del proyecto, para conseguir el objetivo sin complicaciones.

## 

## Arquitectura inicial

Paystats nos ofrece toda la información por medio de consultas en formato json. Gracias a esta forma de recogida de datos, no necesitamos de un software específico como Kafka para recolectar los datos, ya que la aplicación nos ofrece los datos ya de forma ordenada. La información que nos ofrece es de un sandbox con todos los datos del año 2015, por lo que no necesitamos tratar información en tiempo real, y los datos no se van a actualizar una vez publicados.

Por todo lo anterior se va a realizar el procesamiento de toda la información y posteriormente una vez conseguidos los resultados se realizara la publicación en la web. Para ello se realizará una propuesta simple de arquitectura, primero se enlazará el entorno de trabajo en Python con la aplicación, en el que se realizará el procesamiento de los datos, y una vez generados los resultados se generarán archivos de datos, que se incrustarán en el desarrollo web para la generación de las diversas gráficas e informes que nos permitirán responder a las preguntas planteadas.

Jupyter

(Procesamiento de datos)

Paystats

(Origen de datos)

Web

(Visualizacion)

Ficheros de datos

Como podemos ver primero haremos uso de un entorno de desarrollo para trabajar los datos ofrecidos por paystats en python. Haciendo uso de un entorno de trabajo como Jupyter, se ha creado un proyecto de desarrollo en Python que permite realizar la conexión con la aplicación, desde el entorno de trabajo podemos conseguir todos los datos en formato json, se realiza el procesamiento y una vez obtenidos los datos necesarios para representar la información, estos son almacenados en archivos planos.

El segundo paso es la representación de la información procesada, para ello se ha desarrollado una web que permita visualizar de forma clara todos los datos. Primero se ha escogido una plantilla base con un estilo limpio y claro, a partir de ello se han realizado todas las modificaciones y se ha expandido la web para albergar todos los datos. Los datos se recogen de los ficheros de salida y se incrustan en la web, de forma que toda la ejecución se realiza desde el navegador. Para que esta web sea accesible se ha realizado el despliegue y hosting de la web gracias a Firebase.

# Desarrollo de la propuesta.

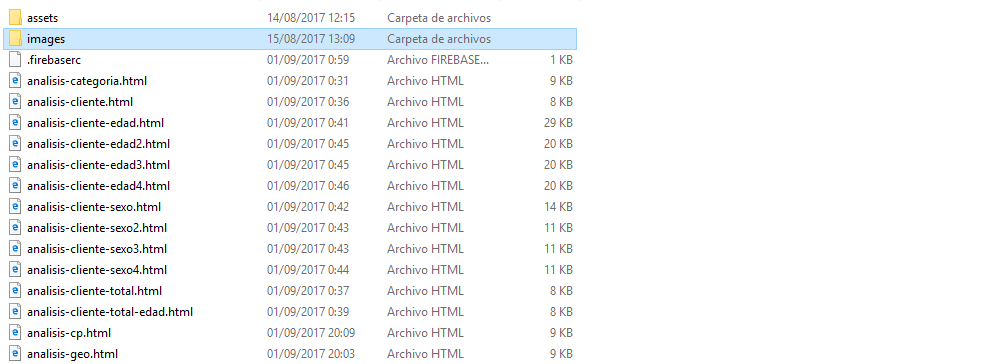
## Análisis detallado

(Desarrollo Python)

## Codificación Web

Para la visualización de los resultados obtenidos en el análisis de datos, se ha desarrollado una web basada en Html5. Para ello nos hemos basado en recursos abiertos que nos permiten obtener un resultado visual limpio, claro y efectivo para mostrar los resultados. Primero hemos partido de una plantilla web ofrecida por Html5up (<https://html5up.net/>), se ha escogido la plantilla Verti, que nos facilita todos los estilos principales de la web. Por otra parte se ha escogido Google Charts como librería para la generación de gráficos (<https://developers.google.com/chart/interactive/docs/quick_start>). Con la ayuda de estos dos recursos se ha desarrollado la web disponible desde cualquier navegador: <https://appbbva-9b62b.firebaseapp.com/>

La web está estructurada en diferentes páginas, las cuales son referenciadas para mostrar todos los resultados obtenidos, además disponemos de los recursos para los estilos de la web, las imágenes utilizadas y archivos de configuración.



En todas las páginas disponemos de un menú que nos permite navegar por las diferentes páginas de la web.



La web está distribuida para ver de forma clara los resultados, dispone de una estructura base de páginas, desde la página inicial, al resumen o la página de contacto. Todos los resultados se han englobado en la sección de análisis. Desde un menú desplegable tenemos acceso a cada una de las secciones, en cada una de ellas además de la descripción de los resultados y otros elementos web, se ha incluido un gráfico explicativo adecuado para la descripción de los datos.

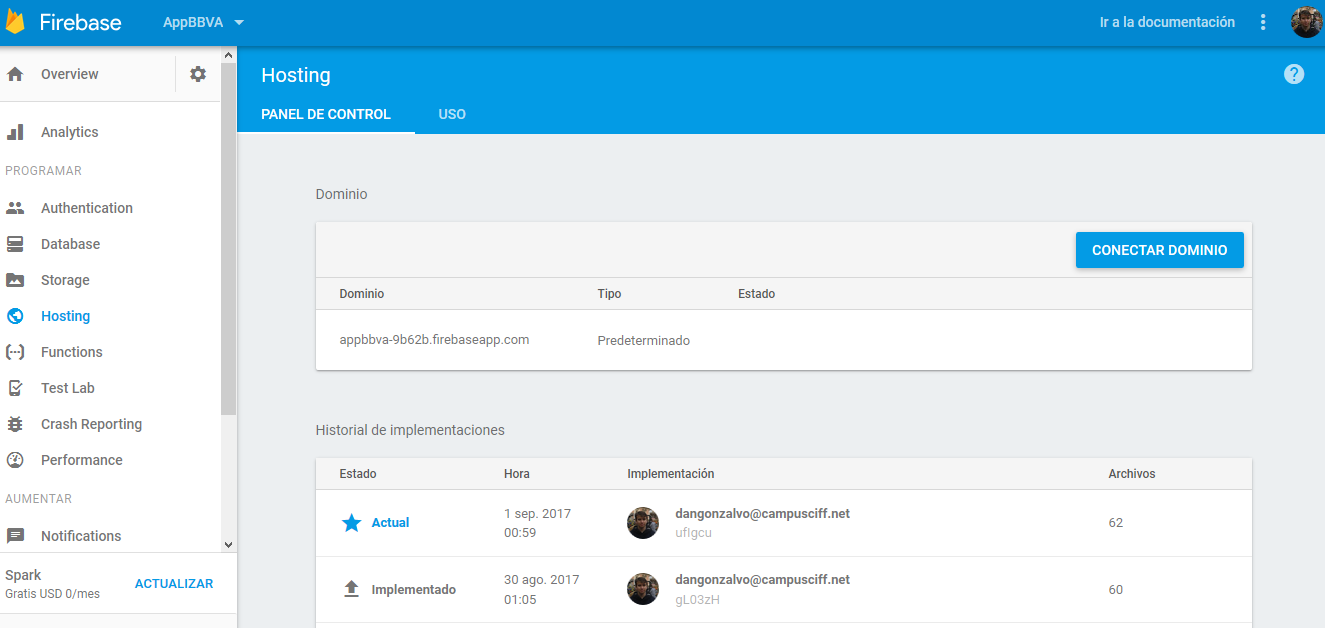


En el cuerpo de la página hacemos referencia al gráfico generado, este puede ser un gráfico de área como el del ejemplo, o cualquier otro tipo de gráfico, el desarrollo será similar, por último añadimos los elementos necesarios para ver los resultados de forma clara. A continuación veremos de forma detallada cada uno de los gráficos y análisis generados.

## Visualización

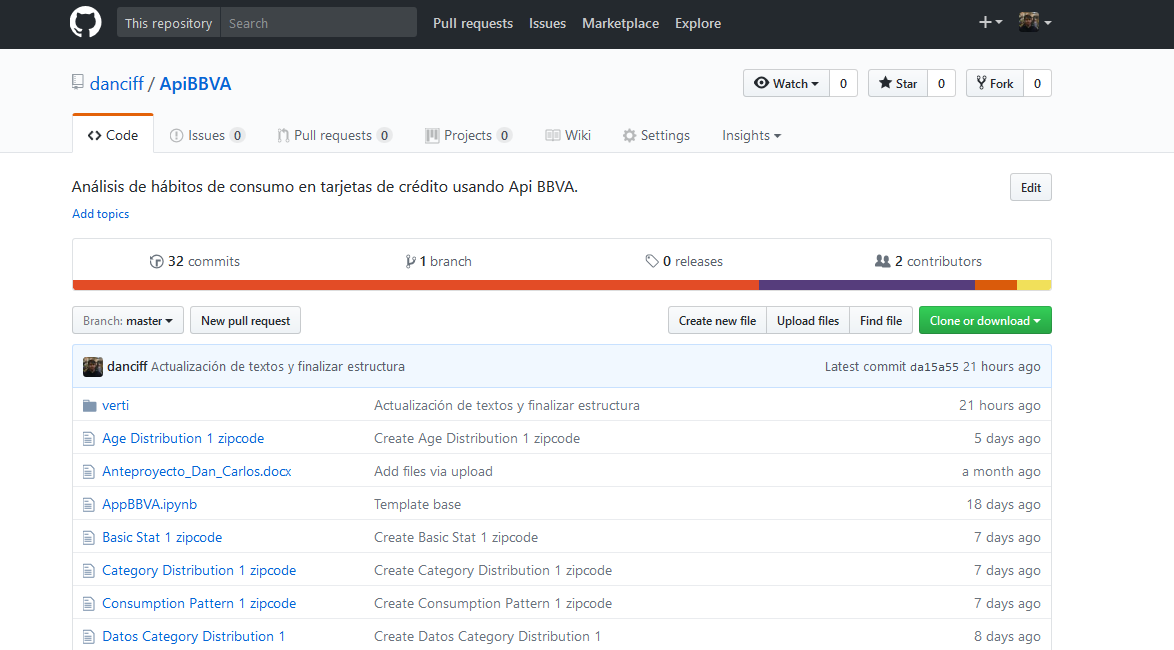
## Implementación Web

Todo el proyecto se ha desplegado gracias a Firebase. Nos permite desplegar nuestra web para la visualización de los resultados, y que esta sea accesible por cualquier usuario desde su navegador.



## Control de versiones

Todo el proyecto está disponible desde Github, se ha utilizado Git como repositorio de toda la codificación desarrollada y de la documentación necesaria: <https://github.com/danciff/ApiBBVA>



## Arquitectura Online

Paystats también nos ofrece la opción de tener acceso a todos los datos posteriores al año 2015 hasta la actualidad. Si se realiza un acuerdo comercial con BBVA podemos tener acceso a estos nuevos datos, y la información se actualizaría de forma semanal. En este caso habría que tener en cuenta esta actualización de los datos por lo que habría que implementar una arquitectura más ambiciosa.

Aunque no se trata de una actualización en tiempo real, sí que habría que hacer una actualización de forma periódica por lo que sería adecuado usar una arquitectura de tipo Kappa. Como en el desarrollo realizado la información la ofrece directamente Paystats por lo que no sería necesario un recolector de información, pero el procesamiento sí que sería necesario realizarlo de forma automática con cada actualización de los datos, y también tendríamos que conectar de forma directa la web con los resultados obtenidos.

Cassandra

(Almacenamiento de datos)

Spark

(Procesamiento de datos)

Paystats

(Origen de datos)

Web

(Visualizacion)

Consultas

En esta arquitectura haríamos uso directamente de la tecnología Spark, como sistema de procesamiento, gracias a PySpark podemos directamente migrar el desarrollo en Python realizado, lo que nos permitirá procesar los datos de Paystats, los resultados en lugar de ser ofrecidos en ficheros, tendrán que ser almacenados en una base de datos como Cassandra. Para recoger los datos podemos hacer uso de servicios Rest, que puedan ser leídos desde la web. Igualmente habría que modificar la web para recepcionar los datos desde servicios rest. Además habría que realizar el hosting no solo de la web sino del procesamiento y almacenamiento.

# Conclusiones.

# Bibliografía.

<https://www.bbvaapimarket.com/products/paystats>

<https://bbvaopen4u.com/es/>

<https://www.bbva.com/>