**Versionamiento**

Debido a la inexperiencia en cuanto a la modularización del código, las funciones implementadas en la iteración número 1 y 2 no tenían un contexto fácil de desenredar para separar en módulos. Esto eso imposible por cuestiones de tiempo adoptar una estrategia formal de versionamiento con ramas.

Se improvisó una especie de protocolo grupal cuyo objetivo fue el de mantener una rama principal actualizada con el código estable más reciente producido por el equipo. Las principales convenciones para lograrlo fueron las siguientes:

* Cada miembro tiene su propia rama para trabajar.
* Al master solo se suben versiones estables (compilables) y que en caso de estar incompletas no intervienen con el trabajo de otros compañeros.
* Las fusiones con la rama master no deben pasar más de un día.
* Se recomiendan los commits pequeños a la rama master.
* Al momento de hacer fusión con el master se debe seguir el siguiente procedimiento:
  + Hacer commit y push de los cambios a la rama personal, para evitar que se pierdan los avances en el proceso de merge.
  + Hacer checkout a la rama master.
  + Hacer pull de la rama master, para traer la versión más nueva que haya sido subida por los otros miembros.
  + Hacer merge con la rama personal y resolver todos los conflictos que se puedan presentar.
  + Correr la aplicación y probar las partes que se modificaron.
  + Hacer push a la rama master
  + Hacer checkout a la rama personal de nuevo, esto para evitar directamente sobre la rama master.
* Continuas actualizaciones de la rama personal respecto a la rama master, con el fin de traer cambios de los compañeros, no para agregar cambios.

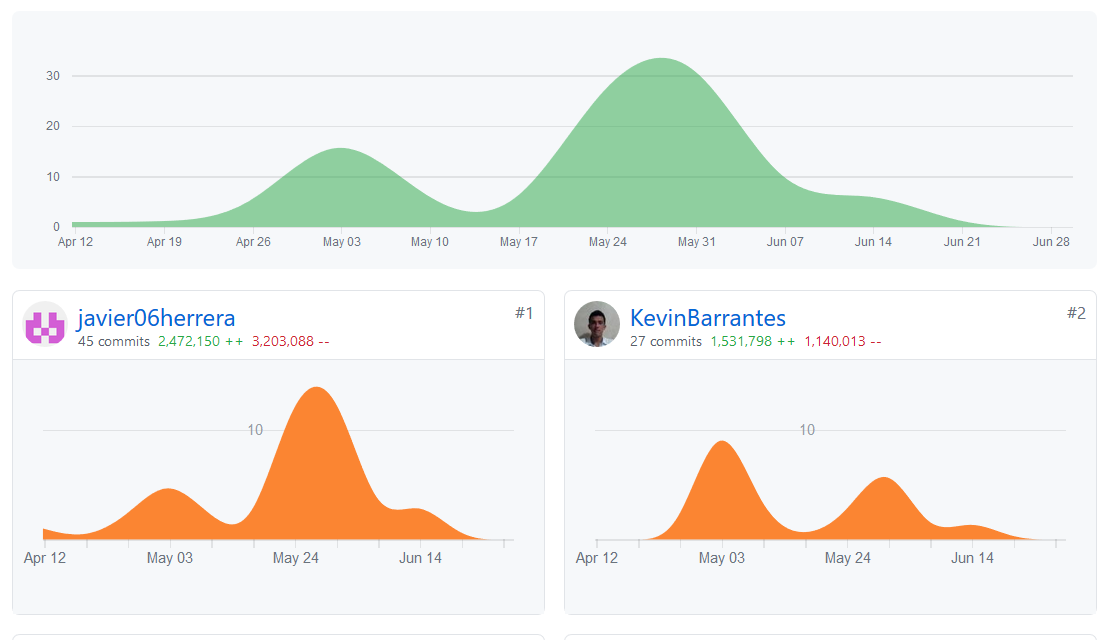
Al final por el tamaño del equipo estas sencillas reglas permitieron una gran fluidez en los procesos de fusión pues generalmente los conflictos más complicados que se encontraron fueron de dos o tres líneas. En cuanto a errores causados por incompatibilidades de las funcionalidades entre las implementaciones de los miembros fueron muy pocos. Las razones se encuentran en dos motivos, el primero es que hacer continuos commits hacía fácil detectar de dónde provenía un error pues el rango de búsqueda era mucho menor, y el segundo es que aunque el código no estaba modularizado las funcionalidades e implementar para esta iteración sí estaban lo suficientemente aisladas como para evitar conflictos entre el trabajo de diferentes miembros pues cada uno se encargo de historias específicas y cuando hubo colaboración fue por pair programming, solo uno subía el código.

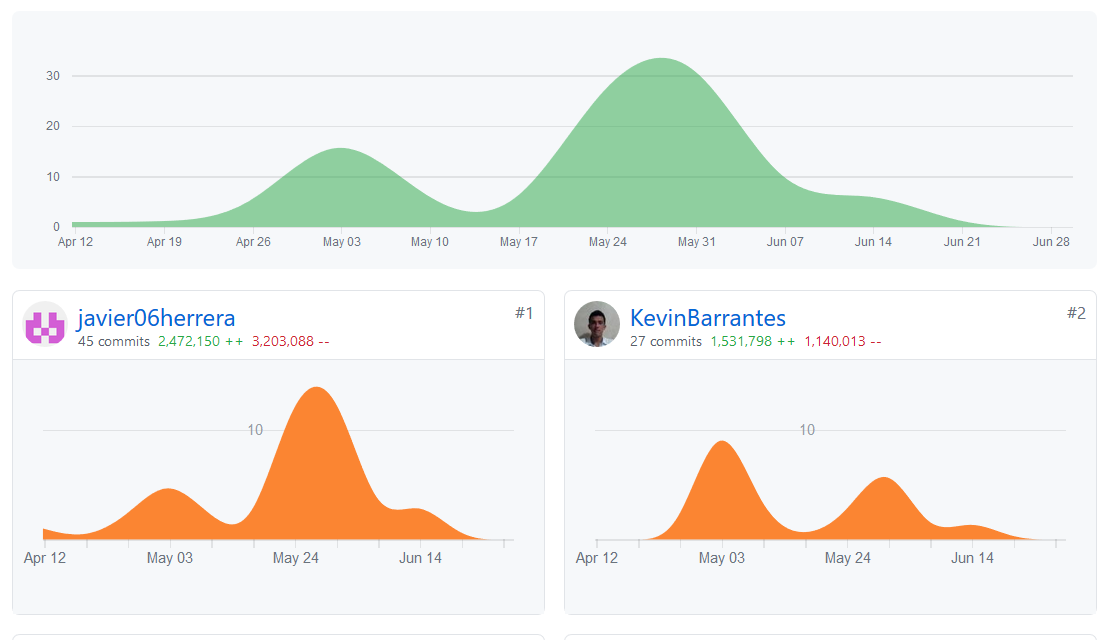
Estadísticas

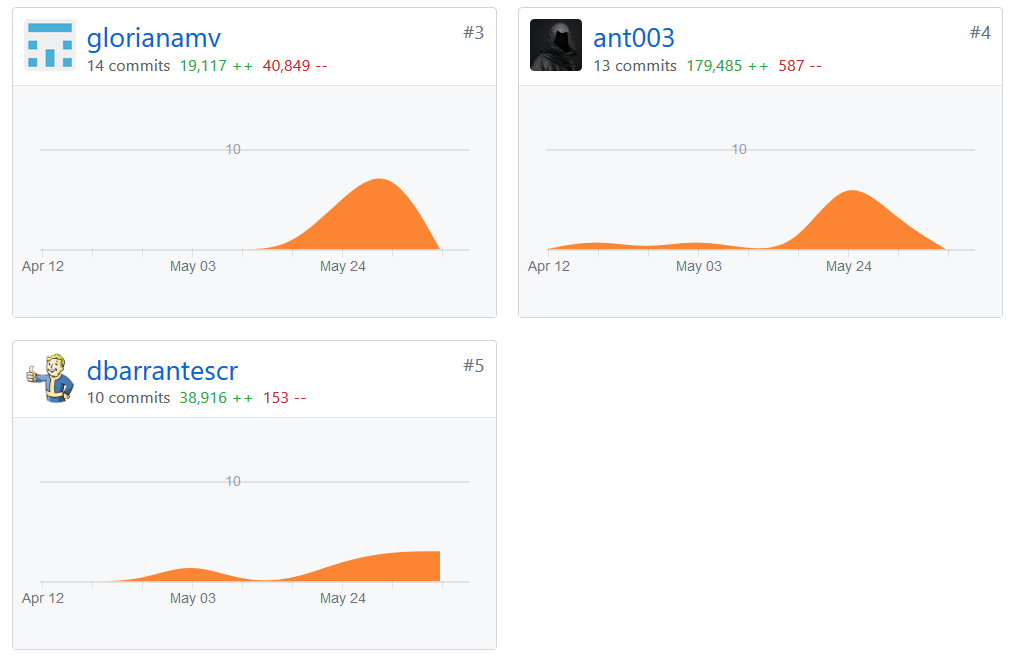
It2: It3:

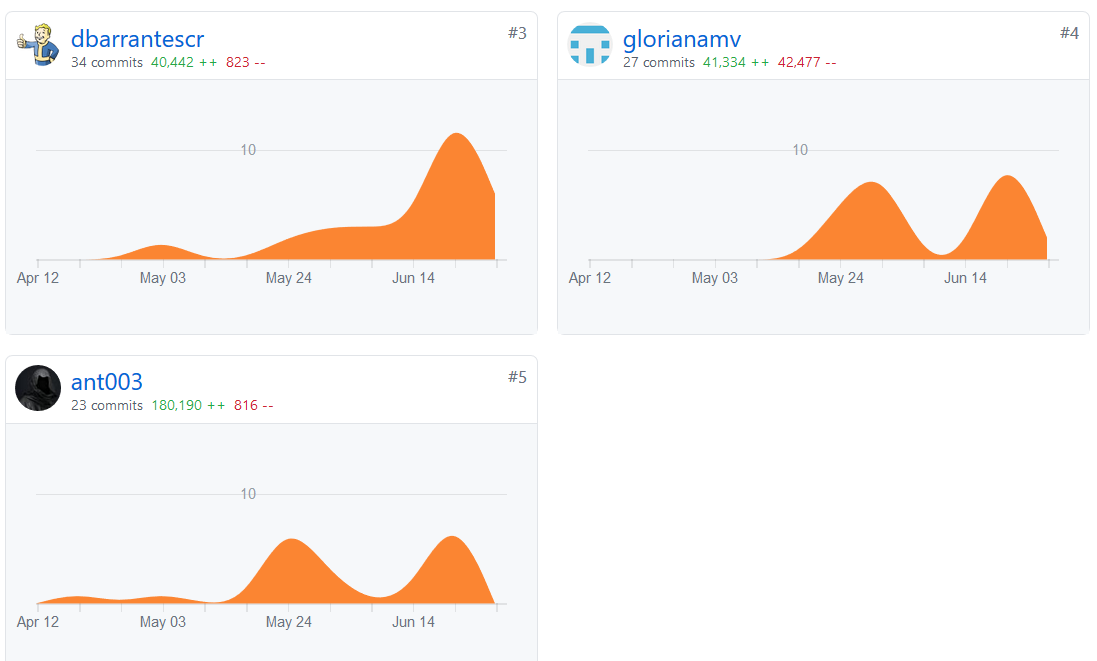
Contribuyentes General:









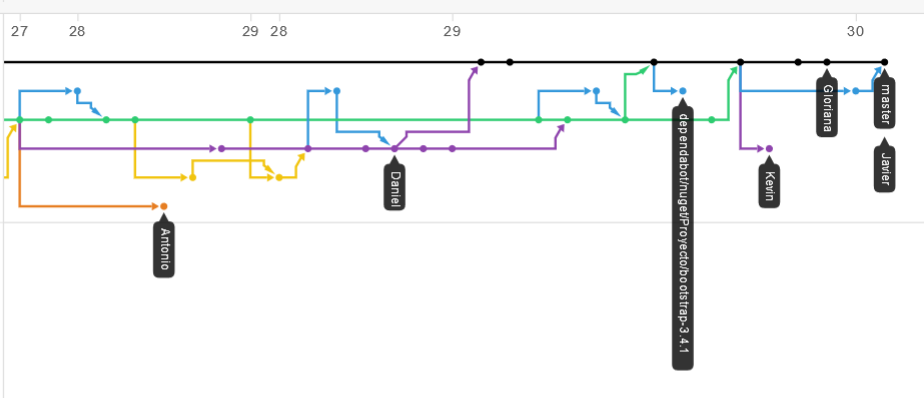


Tráfico



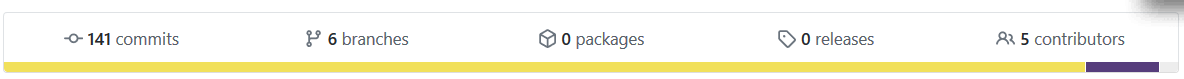
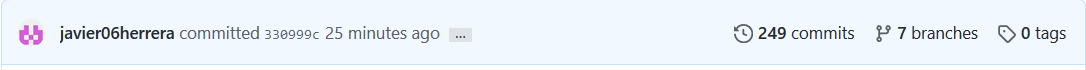


Diagrama de ramificación



No se tiene un diagrama de la iteración pasada, tampoco sería muy razonable compararlos pues solo se pueden mostrar secciones para espacios de tiempo relativamente cortos lo que hace difícil un análisis general del comportamiento de las ramas y poder compararlo con la iteración pasada.

Estadísticas rama master



En general estas estadísticas demuestran un gran crecimiento en el uso del repositorio por parte de todos los miembros.