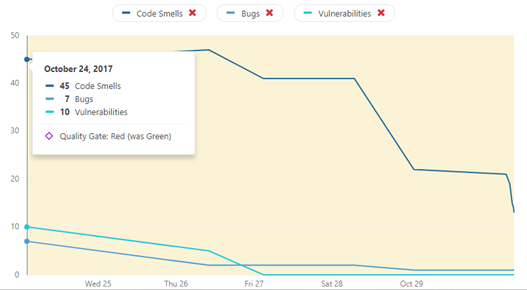
Pràctica 3: SonarQube

Una vegada corregits tots els errors o mals usos de projecte “Petstore”, anem a fer un recorregut per les diferents dades que ens proporciona sonarQube intentant entendre els perquès d'aquestes.

Inicialment, tal com era d’esperar, doncs d’això es tractava aquest treball, el projecte “Petstore” no passava l’anomenat “Quality Gate”. Això era degut principalment al fet que el codi constava, per una banda, de 45 “*code smells”*, és a dir, mals usos i, per l’altra, de 7 bugs i 10 vulnerabilitats. Consegüentment, les valoracions de la fiabilitat i seguretat, no arribaven a la A i, per tant, ja que s’usen els llindars de qualitat per defecte, es considera que el “Quality Gate” no està passat. D’aquesta manera, no calia ni mirar la cobertura que ha de ser com a mínim 80% i, en el cas del projecte, és del 0%. Tanmateix, això forma part del següent treball a realitzar i, per tant, almenys de moment, no ens preocuparem massa. A continuació, es pot observar el gràfic dels primers 5 dies amb les dades ja citades amb anterioritat del projecte:



Tot i que en el gràfic no s’observa prou clarament, a l’inici els “Code Smells” van augmentar, ja que ens estàvem adaptant a l’eina. Tot seguit, però, tal com es pot observar, tots els errors o mals usos dels quals constava el codi van anar disminuint fins al punt que, actualment, hi queden per resoldre 7 “*code smells*” però tant els *bugs* com les vulnerabilitats són 0. D’aquesta manera, les valoracions de fiabilitat, seguretat i sostenibilitat, malgrat els 7 mals usos, són A.

Alguns dels errors més habituals que ens hem anat trobant han sigut la no utilització del *try* i del *catch*¸ la no indicació dels mètodes *override,* problemes a l'hora de declarar atributs i accessos incorrectes als atributs com a conseqüència del mal ús de l’herència. Malgrat tot, quan havíem corregit la major part dels errors, ens vam adonar que el projecte ni compilava ni s’executava correctament a casa d’alguns dels canvis realitzats per alguns dels components del nostre grup. Òbviament, això ens va endarrerir molt. Tal com es pot observar al gràfic que hi ha a continuació, vam passar de 9 “*code smells*” per corregir fins a 29 i això sense tenir en compte els nous bugs i vulnerabilitats sorgides.



Finalment, hem aconseguit tornar a reduir el nombre de “*code smells*” fins a 7, dels quals 4 sorgeixen de la necessitat de crear una excepció pròpia, ja que no es pot generalitzar una classe que estén una excepció de Java. De les 3 restants, dues són com a conseqüència del pas de massa paràmetres i l’última que no hem aconseguit entendre el perquè no es tracta d’un bon ús. La falta de més temps, com a conseqüència dels problemes sorgits, i el fet que vam considerar que els mals usos restants no eren molt crítics, vam decidir deixar-ho en aquesta xifra.

Tot i aquests imprevistos, hem aconseguit reduir força el deute tècnic, és a dir, a les conseqüències mesurades en temps d'usar arquitectures i dissenys deficients o de seguir pràctiques dolentes de desenvolupament de software. A mesura que anàvem modificant el codi, de vegades millorant-lo, altres empitjorant-lo, s’ha vist modificat. De fet, aquest deute, ja que depèn principalment de la quantitat de “*Code Smells*” (45 al principi) dels quals consta el codi, inicialment era d’un dia. Malgrat això, tal com es pot observar a la gràfica que hi ha a continuació, aquest deute va anar disminuint fins que ara és de 2 hores. Gràcies a aquestes dades, és a dir, gràcies a que el rati del deute tècnic és menor al 5%, hem aconseguit que la valoració de la sostenibilitat sigui A.



En conclusió, aquests són els resultats obtinguts al llarg del nostre treball. Hem aconseguit les valoracions màximes en quant a seguretat, sostenibilitat i fiabilitat. No obstant això, ara és important l’estudi de la cobertura, és a dir, el percentatge de codi testejat, ja que mentre que no arribi al 80%, el “Quality Gate” es considerarà que no està passat. Cal remarcar que, tot i que a partir del 80% l’eina ja considera que a escala de qualitat és suficient, cal vigilar que la part del codi no provada no sigui la més utilitzada.