



MY PET CARE

INCEPTION – SEGONA ENTREGA



[Project record track](#) 

Enllaços als recursos: [Taiga](#) 

[Repositori de GitHub](#)

Cognom, Nom	Rol	Contacte (@est.fib.upc.edu)	Drive (@gmail.com)	Taiga i Github
Campos, Xavier	Developer	xavier.campos.diaz	xavicampos99	xavier-campos
Catalan, Oriol	Developer	oriol.catalan.fernandez	catalan.oriol	oricat8
Clemente, Daniel	Developer	daniel.clemente.marin	daniclemente44	danicm47
Del Rey, Santiago	Developer	santiago.del.rey	santi.delrey	santidrj
Hernando, Enric	Developer	enric.hernando	enrichernandopuerta	enrichp
Pinto, Albert	Developer	albert.pinto	apintogil	AlbertPG
Simó, Marc	Developer	marc.simo	sgmarcsg	marcIII
Trius, Álvaro	Master	alvaro.trius	alvarotrius94	AlvarTB

Assignatura: Projecte d'Enginyeria del Software

Professor: Silverio Martínez

Curs: 2019-2020 QP

Facultat d'Informàtica de Barcelona

Universitat Politècnica de Catalunya



ÍNDEX DE CONTINGUTS

1. Sprint master report.....	2
2. Llista de stakeholders.....	4
2.1. Ús	4
2.2. Tema	5
2.3. Tecnologia	6
2.4. Desenvolupament.....	7
3. Temes i històries d'usuari (Product backlog).....	9
3.1. Gestió de mascotes i usuari	9
3.2. Salut de la mascota	10
3.3. Assistència d'exercici.....	11
3.4. Establiments especialitzats.....	11
3.5. Xarxa social en comunitats temàtiques	11
3.6. Interacció amb altres xarxes socials:	12
3.7. Interacció amb un calendari.....	12
3.8. Gestió de medalles i recompenses.....	13
4. Critical mock-ups.....	14
4.1. Registre d'usuari.....	14
4.2. Pàgina principal	15
4.3. Mapa de navegació	16
4.4. Xat	17
5. Arquitectura física.....	18
6. Diagrama UML inicial	19
6.1. Diagrama	19
6.2. Restriccions Textuals.....	20
7. Descripció de la metodologia de treball.....	22

7.1. Metodologia àgil.....	22
7.2. Definició de finalització	22
7.3. Gestió del projecte.....	23
7.4. Convencions	23
7.5. Gitflow.....	25
7.6. Testeig.....	26
7.7. Github actions.....	27
7.8. Codacy	27
7.9. Android studio.....	28

ÍNDIX DE FIGURES

Figura 1: Mock-up de la pantalla d'inici	14
Figura 2: Mock-up de la pàgina principal.....	15
Figura 3: Mock-up del mapa de navegació	16
Figura 4: Mock-up de la pantalla del xat.....	17
Figura 5: Aproximació a la arquitectura física	18
Figura 6: Primera part de l'esquema UML.....	19
Figura 7: Segona part de l'esquema UML.....	20

1. Sprint master report

A aquesta segona iteració de la fase d'inception, hem profunditzat en les funcionalitats de la nostra aplicació. Aquesta iteració va començar el 21 de febrer i va acabar el 6 de Març.

El primer dia ens vam dedicar a avaluar l'Sprint anterior i a decidir el nou Scrum Master. També vam planificar les tasques a realitzar al llarg de aquestes dues setmanes.

La primera meitat de la primera setmana, això és, del 21 de febrer fins al 26 de febrer, vam decidir treballar en els stakeholders i en les històries d'usuari. Per a realitzar la tasca d'identificar els stakeholders, vam dividir-nos en quatre equips de dos components, cadascú encarregat d'un tipus de stakeholder (Ús, Desenvolupament, Tema i Tecnologia). D'altra banda, per a realitzar les històries d'usuari, vam basar-nos en la llista de funcionalitats del document anterior: vam decidir que cada funcionalitat en seria un Tema i vam repartir-los entre tots els membres del grup. El 26 de febrer vam acordar reunir-nos en horari fora de classe per revisar-ho tot i assignar els punts d'història corresponents mitjançant el Planning Poker.


El dia 26 de febrer ens vam reunir amb la intenció de revisar el treball realitzat i fer el Planning Poker. Per desgràcia, d'una banda, algunes de les històries d'usuari eren incorrectes i, d'altra banda, ens va faltar temps per acabar les correctes. Per això, vam decidir posposar la resta del Planning Poker per al dia següent: el 28 de febrer. Mentrestant, cada membre del grup estaria responsabilitzat a fer els canvis necessaris a les seves històries d'usuari i stakeholders.

En aquesta segona reunió també vam acordar eliminar un tema, aquest és, el de la botiga de productes especialitzats: havíem estat estudiant-lo durant aquests dies i ens vam adonar que era una funcionalitat massa complexa com per a implementar-la en una primera *release* de la aplicació. Vam decidir que aquesta seria implementada en un possible futur. Per últim, vam fer un equip de dues persones que van començar a treballar en els mock-ups, aquests havien d'estar acabats per al dimecres 4 de Març.

El divendres 28 de febrer vam dedicar-lo majoritàriament a realitzar el Planning Poker. Un cop finalitzat, ens vam planificar per a la setmana següent. Els stakeholders els vam considerar finalitzats. Ens vam adonar que ens mancaven els criteris

d'acceptació a les històries d'usuari, per això, de cara a la setmana següent, vam decidir que cadascú hauria d'incloure els propis del seu tema. A més, cadascú hauria d'agrupar les seves històries d'usuari de creació, edició, visualització i eliminació en CRUD. Finalment, ens vam dividir el treball pendent en equips de mida diferent: una persona s'encarregaria de l'esquema de la physical architecture, dues s'encarregarien de la presentació al client, altres dues en redactar la working methodology y altres tres en realitzar el diagrama UML, totes aquestes coses a ser revisades el dia 4 de març, juntament amb els mock-ups de la meitat de la setmana anterior. Aquest dia també va sorgir el dubte respecte a un ús concret de la API de Google: volíem fer ús d'algunes funcionalitats per les quals s'hauria de pagar una quantitat de diners, però després d'una consulta amb el professor el 4 de març i una mica d'investigació, ens vam adonar que era factible.

El 4 de març el vam dedicar a revisar el treball realitzat: es van proposar fer canvis al UML i als Mock-ups i va continuar la preparació de la presentació del divendres. Dos membres del grup havien estat treballant des de el primer dia del projecte en el Working Methodology, així que aquest dia i el següent es van dedicar a acabar-ho. El dijous 5 de març cadascú es va dedicar a finalitzar els seus apartats mentre que l'Scrum Master es va dedicar a la redacció d'aquest mateix document.

Finalment, el dia 6 de març vam fer entrega d'aquest document. 

2. Llista de stakeholders

2.1. Ús



-Propietaris de Mascotes:

- Rol: Han d'utilitzar el nostre sistema.
- Objectiu: Gestionar tots els temes relacionats amb les seves mascotes.

-Veterinaris:

- Rol: Donar recomanacions sobre la salut al propietaris mitjançant el nostre sistema.
- Objectiu: Intentar atreure potencials clients a la seva clínica i millorar la seva de les mascotes en general.

-Experts en Alimentació de Mascotes:

- Rol: Donar recomanacions sobre l'alimentació al propietaris mitjançant el nostre sistema.
- Objectiu: Millorar l'alimentació de les mascotes.

-Experts en un cert tipus de Mascotes:

- Rol: Informar als propietaris d'un cert tipus de mascotes sobre la millor forma de cuidar-les i sobre les seves necessitats.
- Objectiu: Millorar el coneixement dels propietaris sobre les necessitats de les seves mascotes.

-Especialistes en Comportament Animal:

- Rol: Informar i aconsellar als propietaris sobre possibles comportaments estranys de les seves mascotes i com tractar-los.
- Objectiu: Intentar captar potencials clients i solucionar problemes derivats del comportament de les mascotes.

-Dissenyador Gràfic:

- Rol: Dissenyar una interfície gràfica adient per a aquest tipus d'aplicació.

- Objectiu: Fer que el sistema compleixi amb els criteris establerts sobre el disseny d'interfícies.

2.2. Tema

-Veterinaris

- Rol: Proveir la informació necessària per al cuidat de les mascotes: des de consells de pràctica diària com a llistes de símptomes per identificar malalties. Atendre a les consultes dels nostres usuaris a través de la app.
- Objectiu: Poder contribuir a la nostra aplicació per garantir el cuidat de les mascotes a tot arreu.

-Propietaris de mascotes:

- Rol: Proveir de experiència en el cuidat de les seves mascotes per tal de millorar les funcionalitats ofertes a les nostres aplicacions. Oferir consells a altres usuaris i obtenir-ne de veus experimentades.
- Objectiu: Fer ús de la nostra aplicació per a donar consell des de la seva experiència, així facilitant el cuidat de les mascotes a tot arreu.

-Experts ensinistradors:

- Rol: Proveir de la informació necessària per a l'adequat ensinistrament d'un gos o un gat, en funció del caràcter i la raça.
- Objectiu: Donar consell als usuaris per a tal de facilitar una millor harmonia entre humans i animals.

-Especialista en llei de protecció de dades:

- Rol: Assessorar en la construcció de la aplicació i de la seva base de dades per a tal de garantir que sigui segura i legal. A més, haurà d'informar promptament de qualsevol canvi legal futur.
- Objectiu: Garantir que les dades guardades al sistema estiguin adequadament protegides, satisfent totes les necessitats legals.

-Organitzacions de protecció d'animals:

- Rol: Assessorar sobre qüestions respecte al correcte tractament d'un gos o un gat. Donar informació sobre situacions d'abús per tal de protegir animals en situacions violentes.
- Objectiu: Conscienciar als usuaris sobre el correcte tractament dels animals per tal de garantir que viuen en unes condicions dignes.

-Experts estadistes:

- Rol: Aportar informació als creadors de l'aplicació sobre les tendències i interessos del públic respecte al cuidat de les mascotes. També aportar informació respecte a l'estat de la nostra aplicació de cara al públic.
- Objectiu: Donar conèixer informació necessària per a tal de que els gestors de l'aplicació puguin contínuament adaptar-se a les noves situacions.

-Experts en nutrició animal:

- Rol: Aportar informació sobre les dietes de les mascotes en funció de la raça o del pes.
- Objectiu: Garantir la correcta nutrició de les mascotes per a tal de preservar-les en salut.

2.3. Tecnologia**-Google:**

- Rol: Proporcionar totes les eines i la documentació necessàries per a desenvolupar l'aplicació.
- Objectiu: Promoure el desenvolupament d'aplicacions per Android i estendre el seu ús per tal de poder augmentar els seus ingressos.

-Facebook:

- Rol: Proporcionar les eines i documentació necessària per tal de realitzar els login en la nostra aplicació i poder compartir dades amb altres de l'empresa.
- Objectiu: Promoure l'ús de les seves aplicacions per a compartir informació a la xarxa i poder augmentar els seus ingressos.

-Twitter:

- Rol: Proporcionar les eines i documentació necessària per tal de poder publicar informació en la seva aplicació.
- Objectiu: Promoure l'ús de les seves aplicacions per a compartir informació a la xarxa i poder augmentar els seus ingressos.

-Microsoft:

- Rol: Proporcionar un sistema de control de versions per desenvolupar, emmagatzemar i compartir el codi de la nostra aplicació.
- Objectiu: Promoure el desenvolupament d'aplicacions open-source.

2.4. Desenvolupament**-Gestor del projecte:**

- Rol: Organitzar la feina que s'ha de realitzar de cara a la següent iteració, coordinar i motivar a l'equip de desenvolupament.
- Objectiu: Portar al dia la feina, tenir controlats els possibles riscos i solucionar possibles problemes que sorgeixen durant la iteració.

-Enginyer de requisits:

- Rol: Fer l'especificació dels requisits.
- Objectiu: Fer un estudi del context i els requisits del sistema.

-Equip disseny del Front-End:

- Rol: Dissenyar la interfície gràfica de l'aplicació per tal de que sigui fàcil d'utilitzar i estètica i desenvolupar-la.
- Objectiu: Desenvolupar la part Front-End del sistema.

-Equip Back-End:

- Rol: Dissenyar la gestió de la base de dades de l'aplicació per tal que sigui funcional, segura i eficient.
- Objectiu: Desenvolupar la part Back-End del sistema.

-Equip de manteniment:

- Rol: Mantenir l'aplicació estable i en funcionament.
- Objectiu: Fer que el sistema funcioni correctament i solucionar ràpidament els possibles problemes que puguin sorgir.

-Programadors Sèniors i Juniors:

- Rol: Programar les diferents històries d'usuari del projecte. En el cas dels programadors sèniors, assessorar i ajudar als programadors juniors.
- Objectiu: Fer que el sistema tingui les funcionalitats correctament implementades i que l'aplicació funcioni correctament.

-Testers

- Rol: Fer tests de les diferents funcionalitats del sistema.
- Objectiu: Trobar els errors de les funcionalitats del sistema i no passar per alt cap bug ni cap petit problema.

3. Temes i històries d'usuari (Product backlog)

(Per a consultar tota la informació referent a aquest apartat, si us plau, consulteu el link a la pàgina del Taiga proporcionada al principi d'aquest document)

3.1. Gestió de mascotes i usuari



Gestió del compte de l'usuari:	Gestió de les dades de l'usuari:	Gestió de les mascotes de l'usuari:	Gestió de la informació general de la mascota:
Alta usuari.	Modificar Nom d'usuari.	Alta mascota.	Modificar Nom Mascota.
Baixa usuari.	Modificar Correu Electrònic.	Baixa Mascota.	Modificar Sexe Mascota.
Modificar idioma del sistema.	Modificar Contrasenya.	Consulta Mascotes.	Modificar Raça Mascota.
	CRUD Imatge de Perfil.		Modificar Data de Naixement.
			CRUD imatge de perfil per a la mascota.

3.2. Salut de la mascota

Informació bàsica	Alimentació:	Higiene:	Veterinari:
Afegir dades salut mascota.	Afegir àpat.	Afegir rentat.	Consultar pròxima visita.
Modificar dades salut mascota.	Modificar àpat.	Consultar pròxim rentat.	Consultar seguiment medicació.
Crear nou perfil mèdic mascota.	Eliminar àpat.	Consultar historial de rentats.	Consultar historial mèdic.
Eliminar perfil mèdic mascota.	Calcular kcal aconsellades.		Afegir visita veterinari.
Afegir pes.	Consultar kcal Consumides.		Afegir medicació.
			Eliminar visita veterinari.
			Eliminar medicació.
			Modificar medicació.
			Modificar visita veterinari.

3.3. Assistència d'exercici

Gestió de les rutes de passeig:	Control de l'exercici realitzat:
Realitzar passeig.	Gestionar l'exercici realitzat.
Consultar ruta de passeig.	Compartir exercici realitzat.
Eliminar ruta de passeig.	Eliminar els registres més antics periòdicament.
Compartir ruta de passeig.	

3.4. Establiments especialitzats

Localització d'establiments especialitzats:	Valoració d'establiments especialitzats:
Veure establiments propers.	Crear valoració.
Filtrar la visualització dels establiments.	Modificar valoració.
Mostrar la ruta més ràpida.	Eliminar valoració.

3.5. Xarxa social en comunitats temàtiques

Gestió de continguts:	Gestió de grups:
Pujar, veure, comentar i eliminar contingut/fòrum (CRUD).	Crear/Esborrar/Editar grup (CRUD).
Donar like/dislike al contingut/fòrum.	Subscriure's al grup.
Denunciar contingut/fòrum.	Desubscriure's al grup.
	Rebre notificacions del grup

3.6. Interacció amb altres xarxes socials:

Facebook:	Twitter:	Instagram:	WhatsApp:
Iniciar sessió.	Iniciar sessió.	Iniciar sessió.	Compartir fotos de la galeria.
Tancar sessió.	Tancar sessió.	Tancar sessió.	Compartir aplicació.
Compartir fotos de la galeria.	Compartir fotos de la galeria.	Compartir fotos de la galeria.	Compartir publicacions de fòrums.
Compartir aplicació.	Compartir aplicació.	Compartir aplicació.	Compartir informació de la teva mascota.
Compartir publicacions de fòrums.	Compartir publicacions de fòrums.	Compartir publicacions de fòrums.	
Compartir informació de la teva mascota.	Compartir informació de la teva mascota.	Compartir informació de la teva mascota.	

3.7. Interacció amb un calendari

Avisos personals:	Avisos automatitzats:	Notificacions a l'usuari:
CRUD avís personal.	CRUD avís generat.	CRUD notificació periòdica.

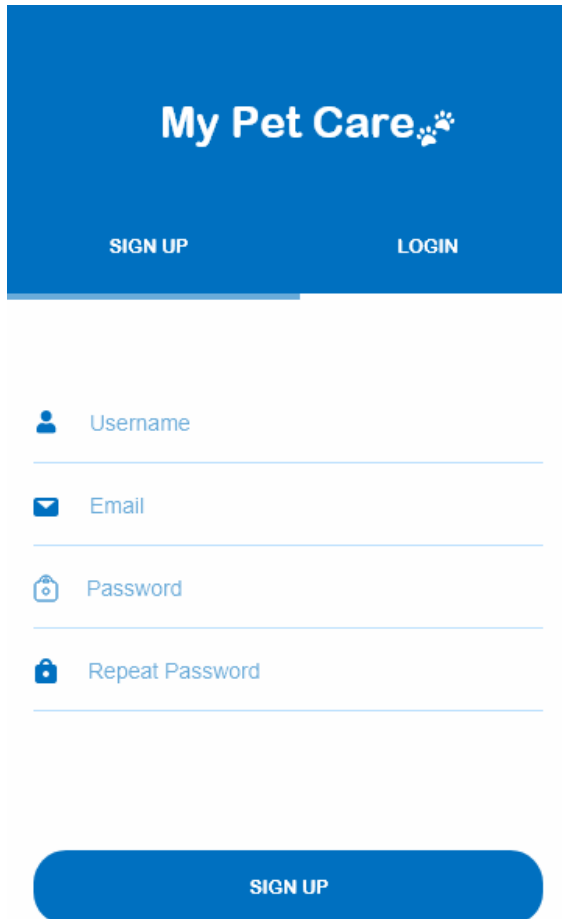
3.8. Gestió de medalles i recompenses

Control del progres de les medalles:	Consultar les medalles obtingudes:	Control medalles disponibles:
Consultar requisits i progrés d'una medalla.	Consultar medalles.	CRUD medalla.
Actualitzar progrés de les medalles.		

4. Critical mock-ups

Per a tal de fer més clara la idea que tenim, hem decidit crear una sèrie de mock-ups. No són definitius, però serviran per acompanyar la presentació que farem al client per a fer-li més partícip de la nostra visió.

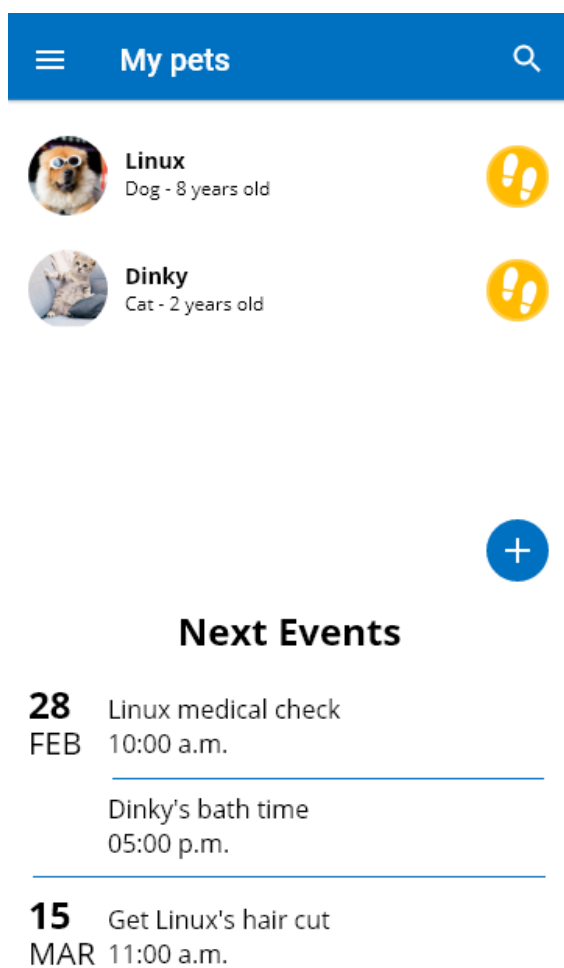
4.1. Registre d'usuari

The mock-up shows a registration form for 'My Pet Care'. At the top, there's a blue header with the app's name and a paw print icon. Below the header, there are two buttons: 'SIGN UP' and 'LOGIN'. The main form area has four input fields, each with an icon and a label: a person icon for 'Username', an envelope icon for 'Email', a key icon for 'Password', and a padlock icon for 'Repeat Password'. At the bottom of the form is a large blue button labeled 'SIGN UP'.

En la pantalla del registre d'un usuari, es possible crear un compte en la nostra aplicació. En concret, ha d'introduir el nom d'usuari desitjat, un correu electrònic i una contrasenya, la qual ha d'introduir dos cops. En quan es premi el botó de confirmació, si el correu no està registrat amb un altre compte i , en el cas que fos així, es mostrarà un missatge d'error. Altrament, s'iniciarà l'alta de l'usuari al sistema.

Figura 1: Mock-up de la pantalla d'inici

4.2. Pàgina principal

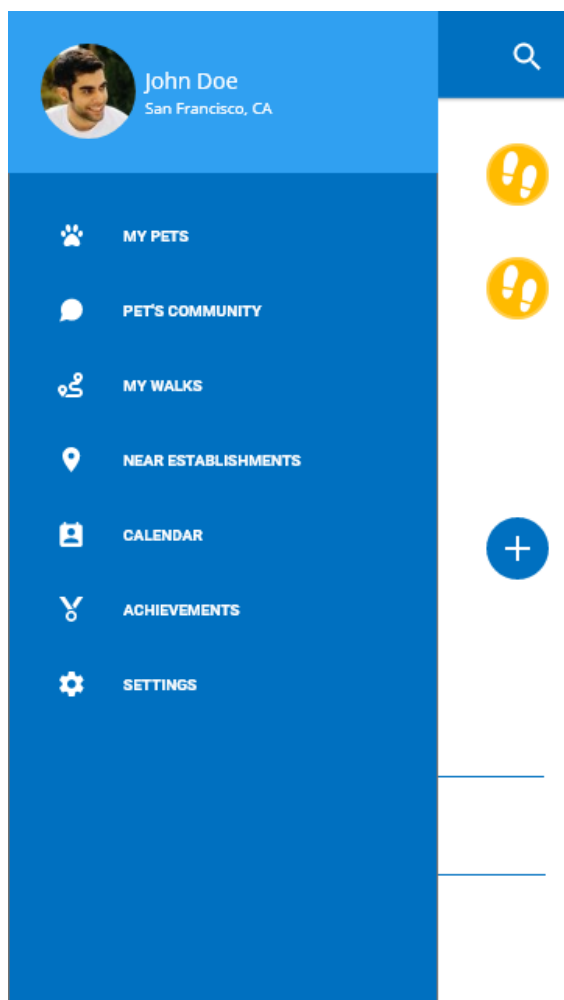


En la pantalla principal de l'aplicació, l'usuari pot veure les mascotes que te registrades en el sistema. Per a cadascuna d'aquestes, es mostra el seu nom, una imatge, la seva raça i la seva edat. A més, al costat de cada mascota es mostra una icona per iniciar un passeig amb elles directament.

En la part inferior de la pantalla, podem veure els pròxims esdeveniments registrats en el calendari referents a les mascotes. En prémer un d'ells, es mostra informació detallada sobre aquests esdeveniments.

Figura 2: Mock-up de la pàgina principal

4.3. Mapa de navegació



Per tal d'accedir a les diferents funcionalitats de l'aplicació, aquesta disposa d'un mapa de navegació, el qual és accessible des de totes les pantalles de l'aplicació. En seleccionar una funcionalitat, el sistema mostra la seva pantalla corresponent.

Figura 3: Mock-up del mapa de navegació

4.4. Xat

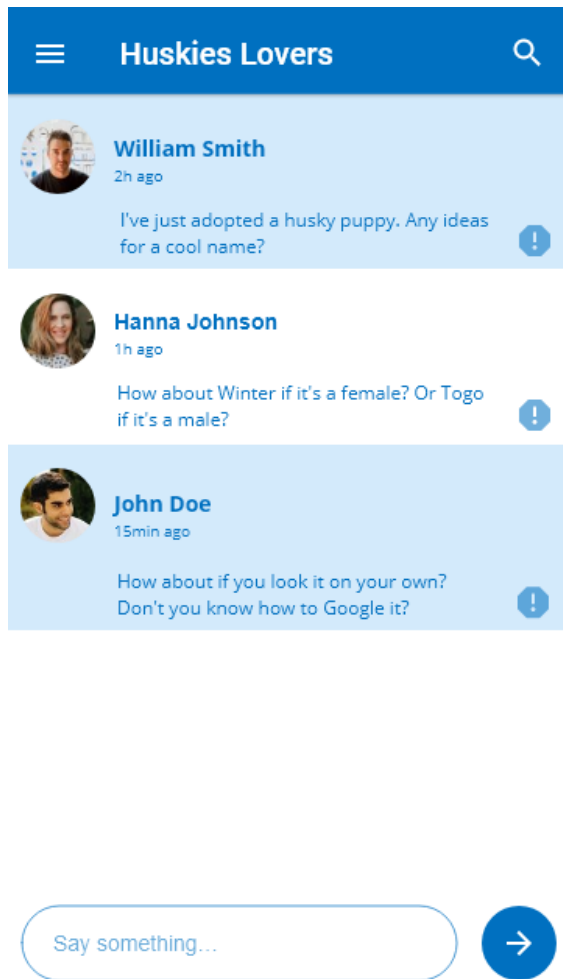



Figura 4: Mock-up de la pantalla del xat

En el xat  incorporat en la nostra comunitat social, els usuaris podem realitzar una publicació sobre un tema i/o assumpte concret. En quan un usuari l'inicia, publica el missatge inicial i, aleshores, la resta d'usuari poden participar en la conversa. A més, es disposa d'un sistema de report per tal de notificar als administradors de l'aplicació si algun usuari té un comportament no acceptable.



5. Arquitectura física

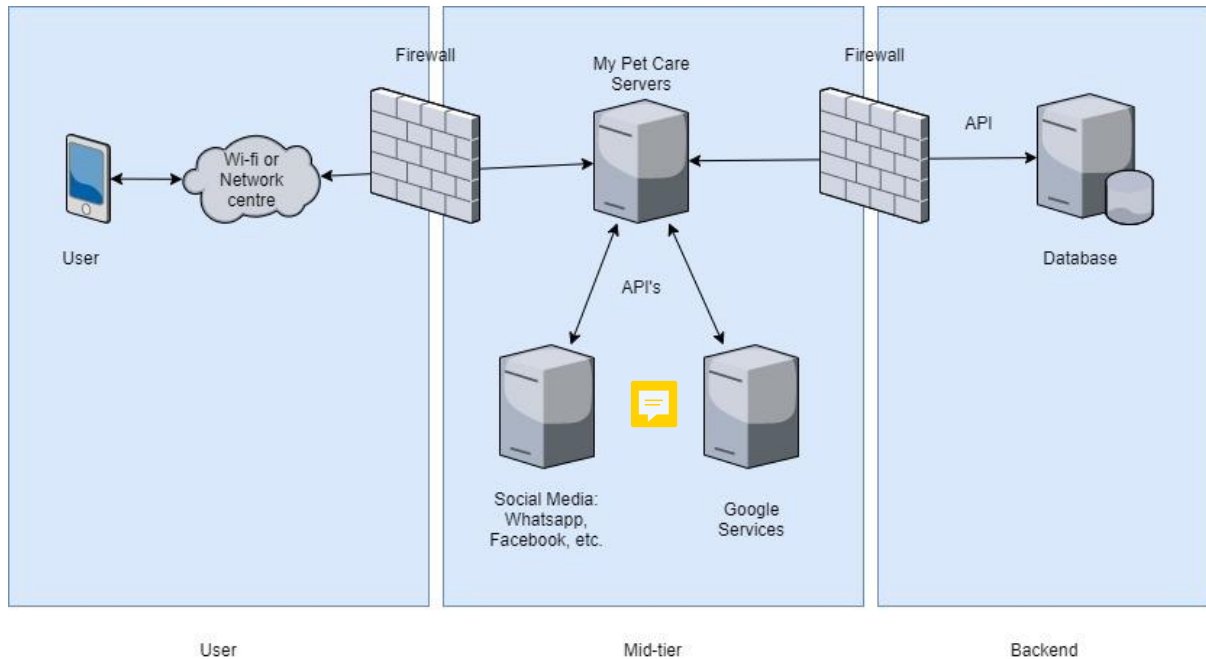



Figura 5: Aproximació a la arquitectura física

Com es pot observar, l'usuari es connectarà mitjançant wi-fi o qualsevol altre proveïdor de Internet amb els nostres servidors. Des d'aquí, l'usuari podrà fer servir dels serveis My Pet Care.

Per aquells serveis que es requereixi la intervenció d'algun altre servidor (com, per exemple, compartir alguna cosa per xarxes socials o visualitzar el mapa de Google) el nostre servidor contactarà amb els proveïdors corresponents mitjançant unes APIs.

Per a realitzar intercanvis d'informació amb la base de Dades, el nostre servidor farà ús d'una altra API per a contactar amb aquesta.

Finalment, cal fer notar que aquesta estructura està dividida en tres parts (User, Mid-tier, Backend), cadascuna separada pel seu Firewall corresponent. 

6. Diagrama UML inicial

6.1. Diagrama

(Per tal de fer el diagrama UML més visible a aquest document, l'hem dividit en dos parts, que estan connectades per "Pet" i "Event")

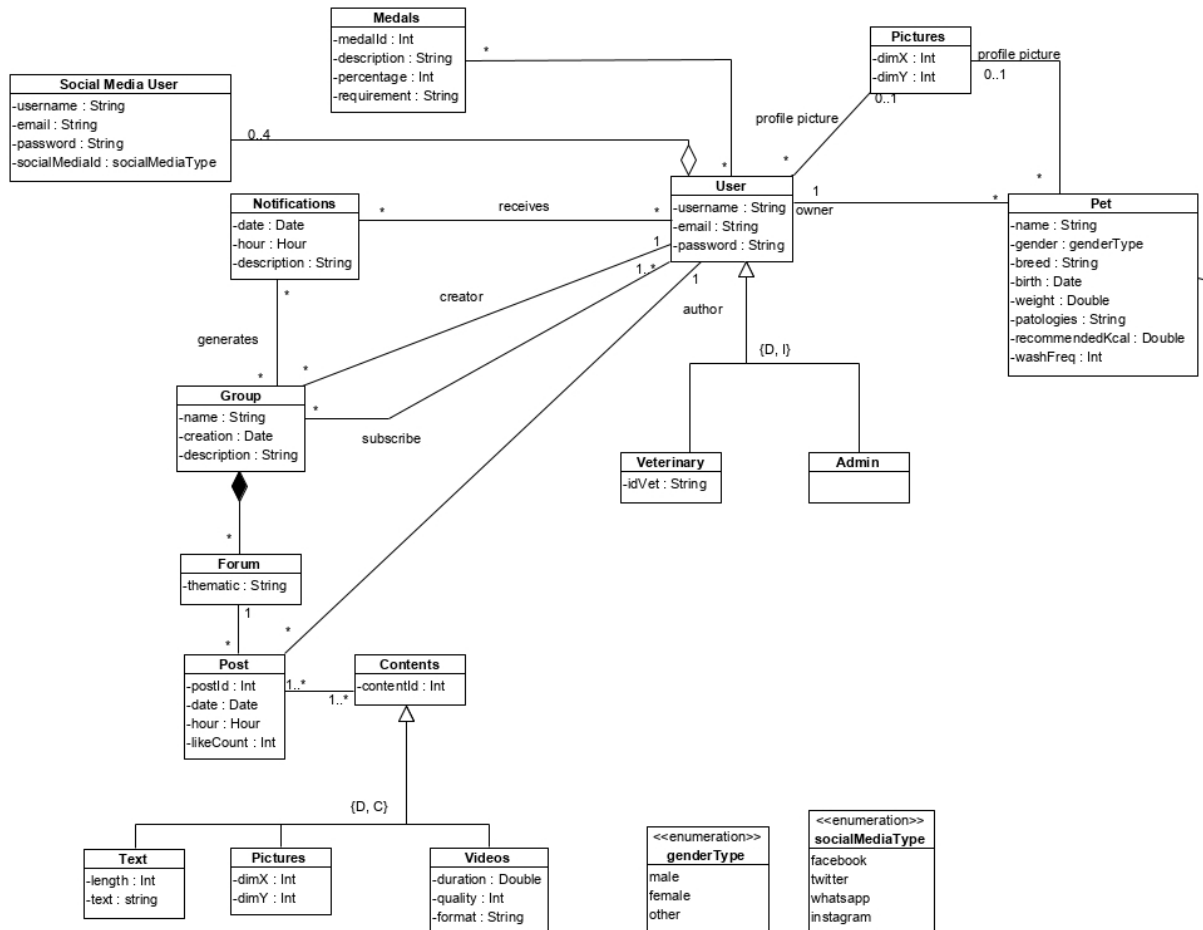


Figura 6: Primera part de l'esquema UML

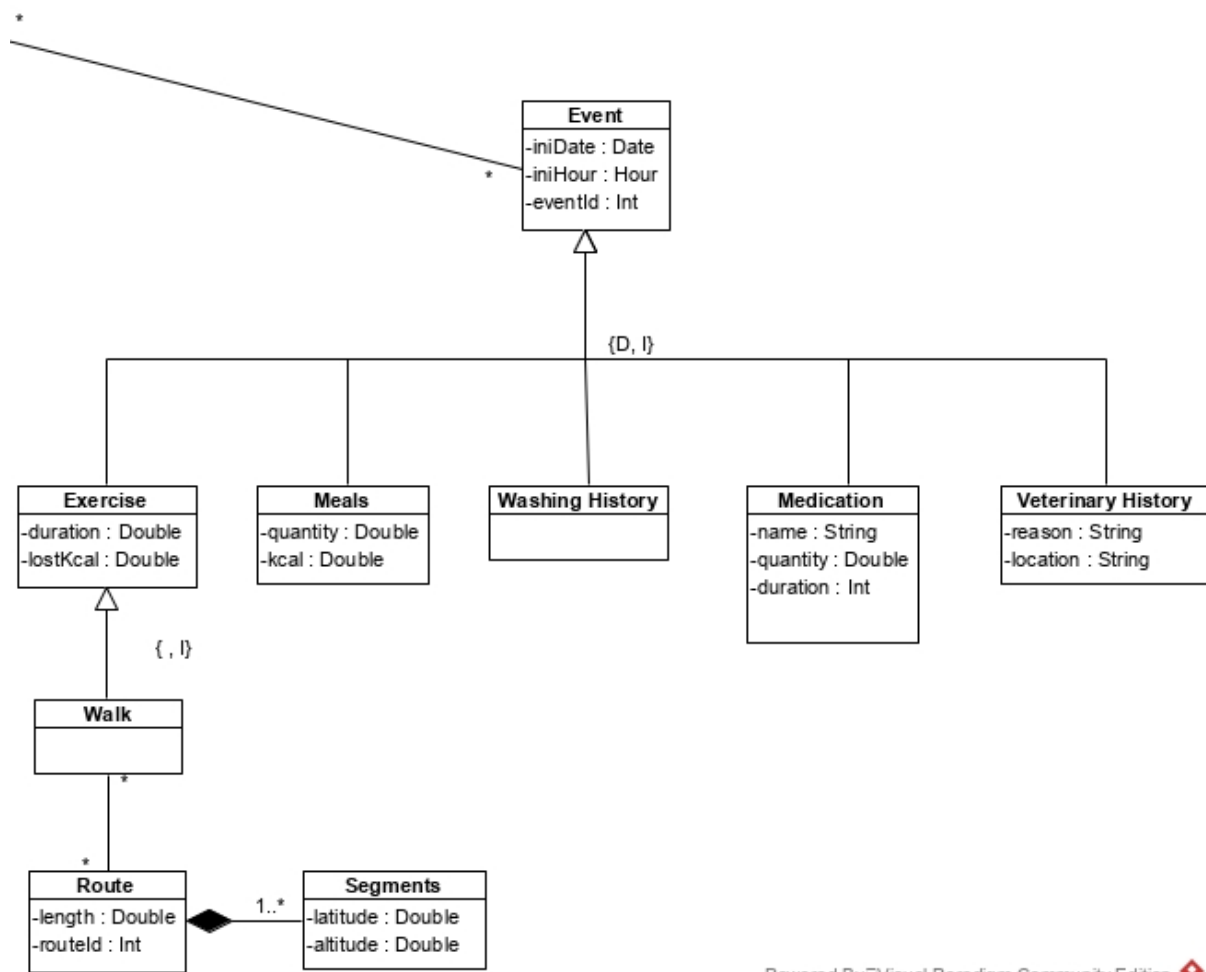


Figura 7: Segona part de l'esquema UML

6.2. Restriccions Textuals

- Claus Externes

- User : username
- Pet : name + owner.username

- Event : eventId
- Contents : contentId
- Route : routeId
- Segment : latitude + altitude
- Group : name
- Forum : thematic + group.name
- Post : postId
- Notifications : group.name + user.username + date + hour
- Social Media User : username
- Medals : medalla


- Tots els integer i els double han de ser ≥ 0 .
- Un usuari només pot ser veterinari si s'ha validat el seu id de veterinari.
- Un usuari administrador pot accedir i modificar tota la informació i continguts del sistema.
- Només podran ser administrador aquells usuaris que formen part de l'equip de desenvolupament del sistema.
- La data d'una notificació ha de ser posterior a la data de creació del grup que l'ha generat.
- Un usuari només pot rebre notificacions d'un grup si és membre del grup.
- Un usuari només pot publicar un post a un fòrum si aquest està subscrit al grup al qual pertany.
- La data d'un post d'un fòrum ha de ser posterior a la data de creació del grup al qual pertany.
- El percentatge d'una medalla ha de ser ≥ 100 .
- La data d'un esdeveniment ha de ser posterior a la data de naixement de la mascota a la qual està associada.

7. Descripció de la metodologia de treball

7.1. Metodologia àgil

Per a realitzar el projecte, hem decidit utilitzar SCRUM, una de les principals metodologies de desenvolupament de software àgils utilitzades actualment. En aquesta, es divideix el desenvolupament del projecte en iteracions o *sprints*, al final de les quals es disposarà d'una versió entregable del sistema.

En aquesta metodologia àgil, es defineixen tres rols destacats. Primerament, el product owner, el qual vetllarà per al compliment de les funcionalitats acordades i, en el nostre cas, serà sempre la mateixa persona. En segon lloc, tenim el scrum master, qui verificarà que s'estigui aplicant la metodologia de treball correctament i redactarà la documentació necessària per la iteració actual. Aquest rol serà rotatiu entre els diferents membres de l'equip. Finalment, els desenvolupadors són els encarregats d'implementar les funcionalitats acordades amb el client.

Per tal de coordinar les nostres activitats, durant les iteracions realitzarem les diferents reunions que estableix aquesta metodologia. Per una banda, al principi de cada iteració, realitzarem una reunió per decidir que realitzarem en aquesta i durant el seu transcurs cada dia que ens trobem en persona indicarem quines tasques hem fet durant la setmana, què farem avui i quins problemes hem tingut. Per l'altre banda, al final de cadascuna de les iteracions realitzarem dues reunions, una amb el client per mostrar les funcionalitats implementades i l'altre per debatre sobre quins aspectes del desenvolupament podríem millorar. 

7.2. Definició de finalització

Per tal de decidir quan una història d'usuari està acabada hem establert un conjunt de criteris definits a continuació:

1. Està definida i conté els seus criteris d'acceptació en el Taiga.
2. Està implementada amb els seus tests necessaris per a complir els seus criteris d'acceptació.
3. Els mètodes de les classes estan documentats.
4. Passa la revisió automàtica i és verificada per com a mínim un altre membre.
5. S'ha afegit el codi a la branca de desenvolupament.

6. Els diagrames de classes estan actualitzats.


7.3. Gestió del projecte

Per tal de realitzar el nostre projecte, hem decidit utilitzar un conjunt d'eines per la gestió d'aquest. Primerament, per tal de controlar les històries d'usuari que hem dissenyat, ens hem decantat per fer servir Taiga, una eina gratuïta específica per a gestionar projectes àgils. Aquesta ens permet definir les èpiques i les seves històries, puntuar-les segons la seva complexitat, organitzar el *product backlog*, planificar els *sprints* i observar l'evolució de l'estat de les històries.

En segon lloc, per tal de gestionar el codi del projecte hem optat per fer servir Github, un sistema de control de versions (VCS). En aquest, utilitzem una metodologia de treball anomenada Gitflow, en la qual s'organitza el codi en branques independents entre elles. A més, aquesta eina és compatible amb les altres eines que utilitzarem en el projecte.

Finalment, per tal de controlar les hores de dedicació personal al projecte, hem creat un full de càlcul anomenat *Project record track*. En aquest, hem d'afegir les diferents activitats vinculades al projecte que anem realitzant, com per exemple, la realització del codi o la redacció de la documentació.

7.4. Convencions

Per tal de gestionar correctament les iteracions hem definit un conjunt de convencions de nomenclatura.¹ Els convenis que tenen un asterisc entre parèntesis “(*)” són verificades pel Codacy: 

- Tot allò que no sigui documentació o un comentari s'ha de realitzar en anglès.
- Els noms de les branques relacionades amb noves funcionalitats s'anomenaran *feature_user_story_name*.

¹ Per consultar tots els convenis establerts, consulteu el següent [enllaç](#).

- Les *release* tindran un nom en el format X.Y.Z, començant en la 1.0.0, tal que:
 - Per a cada nova *release* s'incrementa la X.
 - Per a cada *hotfix* solucionat s'incrementa la Z.
 - Si la solució d'un *hotfix* ha provocat un gran canvi en l'aplicació, s'incrementa la Y.
- Els *pull request* realitzats s'anomenaran com la branca des d'on provenen, amb una breu descripció si es creu convenient.
- Els missatges dels commits s'escriuen en present.
- Exceptuant el project record track, cal iniciar sessió en totes les eines amb el compte de Github, si és possible.
- Els noms de les classes han de seguir un dels següents patrons, en funció de quin sigui el seu contingut:
 - Activitat: NomActivity
 - Fragment: NomFragment
 - Gateway: NomGateway
 - Vista (component): NomView
 - Altres: poden tenir el nom que es consideri oportú.
- En tots els fitxers en java s'utilitzarà camel case en tots els noms (*); malgrat tot, en xml s'escriurà tot en minúscula separant les paraules amb guions baixos, excepte amb els identificadors els quals hauran d'estar amb camel case.
- Els noms de les variables i dels mètodes han de ser significatius.
- Les interfícies no poden començar amb el prefix "I".
- Els identificadors dels components han de començar amb un prefix significatiu, els quals s'aniran concretant sota demanda. Per exemple, btnNom pels botons o lblNom per les etiquetes de text.
- Les funcions no poden tenir més de 15 línies de codi, sense comptar els comentaris (*).
- Les funcions no poden tenir més de 5 paràmetres (*).
- L'idioma utilitzat en el codi és l'anglès, excepte el contingut dels elements del fitxer de recursos Strings.xml amb un locale assignat.

- Les diferents pantalles de l'aplicació, excepte el log in, han de ser implementades utilitzant fragments. Per tal de poder comprovar el funcionament n'hi ha prou amb crear una activitat i incloure el fragment, tot i que aquesta no pot estar en el pull request realitzat.
- S'ha de respectar l'arquitectura en tres capes establerta.
- Per accedir als diferents elements de la interfície gràfica, s'ha de fer servir el View Binding, introduït a Android Studio en la versió 3.6.

7.5. Gitflow

Per tal d'utilitzar el repositori de codi eficientment i evitar problemes d'incompatibilitat de versions, hem decidit utilitzar una metodologia de treball anomenada gitflow. Aquesta estableix en quin tipus de branca del repositori s'ha d'incorporar el codi que hem realitzat i quin és el procediment a seguir en cada cas. En total distingim 5 tipus de branques: *master*, *develop*, *feature*, *release* i *hotfix*.

Primerament, en la branca de *master* es troba el codi de l'aplicació que forma part d'una versió publicada de l'aplicació, és a dir, el codi resultant al final d'una iteració. Per tant, no es pot realitzar cap *commit* que provingui directament de l'entorn de desenvolupament en aquesta branca. En el nostre projecte, hem acordat que l'únic cas on està permès fer un commit sense passar pel procediment habitual és quan incorporem al repositori la documentació d'aquest, la qual es troba únicament en aquesta branca.

Acte seguit, trobem la branca *develop*, la qual prové de la *master*. En aquesta s'aniran incorporant les diferents funcionalitats que es vagin desenvolupant durant el projecte, un cop hagin passat els diferents controls de qualitat establerts. De forma similar a la branca *master*, no es pot realitzar un commit directament en aquesta, sinó cal passar abans per un altre tipus de branca.

A continuació, disposem de les branques de tipus *feature*, les quals contenen el codi de les noves funcionalitats de l'aplicació o que solucionen problemes trobats durant el desenvolupament. Aquestes branques provenen de la branca *develop* i es on es realitzen tots els *commits* des de l'entorn de desenvolupament. D'aquesta manera el codi nou queda aïllat del codi comprovat i potencialment funcional de la branca original. Per tal de poder incorporar aquests canvis a la *develop*, és a dir, realitzar la

fusió entre les branques, és necessari realitzar un *pull request*, en el qual es passaran un seguit de tests d'estil de forma automàtica i una comprovació de compatibilitat entre classes. A més, hem establert que el codi ha de ser verificat totalment per una altre membre del grup, el qual no hagi participat en el desenvolupament d'aquesta part. Malgrat tot, tots els membres poden comentar el codi i suggerir millores d'aquest. Un cop tots els tests han passat i el codi ha estat verificat, es pot realitzar la fusió amb la branca *develop* i, acte seguit, s'eliminarà la branca original.

Quan s'acosta el final de la iteració, es decideix finalitzar la incorporació de noves funcionalitats i iniciar una branca del tipus *release*, la qual prové de la branca *develop*. Totes les funcionalitats que no han estat incorporades a la branca *develop* abans de l'inici d'aquesta nova branca queden fora de la versió actual de l'aplicació. En aquesta, es solucionaran petits errors o *bugs* que s'hagin detectat i es prepararà el codi per a ser lliurat com una versió de l'aplicació. Aquestes comprovacions es realitzaran mitjançant un *pull request* en el qual tots els membres del grup hauran de revisar alguna part del codi per tal de trobar inconsistències o detectar problemes. Un cop ha estat verificat es realitza primer la fusió amb la branca *master* i, acte seguit, amb la *develop*.

Finalment, si un cop s'ha publicat la versió de l'aplicació és detecta algun problema amb l'aplicació, sobretot provinent de les proves amb usuaris reals, s'ha d'iniciar una branca anomenada *hotfix* des de la *master*. Aquesta s'encarrega de solucionar aquests problemes i, acte seguit, s'inicia un *pull request*. Un cop el codi ha estat verificat i s'ha comprovat que aquests han estat solucionats, es realitza la fusió amb la branca *master* i, a continuació, amb la *develop*.

7.6. Testeig


Durant el desenvolupament del projecte, hem de realitzar un conjunt de tests per assegurar la integritat i el correcte funcionament de l'aplicació. Per a dur-los a terme, farem servir tres frameworks diferents: JUnit, el qual permet realitzar tests unitaris, Mockito, que permet realitzar mocks en tests unitaris, i Espresso, que permet realitzar tests instrumentals. L'ús d'aquests frameworks permetrà que puguem aplicar el TDD per a desenvolupar el codi relacionat amb la lògica de l'aplicació.

7.7. Github actions

Per tal d'aplicar els principis de *continuous integration* i *continuous deployment*, farem servir les *pipelines* que ens proporciona Github, ja que és el nostre sistema de control de versions i, per tant, tindrà una millor integració amb el repositori. Aquestes *pipelines* ens proporcionen un mètode per automatitzar la realització de tests, la creació de paquets i muntar o desplegar la nostra aplicació.

Primerament, es defineix el nom que portarà la *pipe* i els esdeveniments que l'activen. Per a cadascun d'aquests, es pot escollir en quina o quines branques es pot iniciar. A continuació, es creen les tasques a realitzar, les quals s'executaran en paral·lel si no s'indica el contrari. Per a la creació de cadascuna d'aquestes s'indica el seu nom i, després, s'especifica en quin o quins sistemes operatius s'executarà, podent escollir entre diferents versions de Linux, mac Os o Windows. Un cop arribats a aquest punt, es defineixen les etapes que ha de seguir la tasca. Aquestes poden ser comandes del terminal o altres accions definides per Github o per altres usuaris.

7.8. Codacy

Per tal de mantenir el nostre codi net i sense *code smells*  utilitzarem una eina de revisió i anàlisi de codi anomenada Codacy. Aquesta eina disposa d'un seguit de regles, les quals pots activar i personalitzar, per tal de crear uns criteris de revisió pel codi. Amb aquestes, s'intenta que tot el codi desenvolupat dins del projecte segueixi els mateixos criteris d'estil, és a dir, que estigui estructurat i definit de la mateixa manera independentment de qui l'hagi escrit.

Primerament, l'eina porta un control dels problemes detectats als codis, com poden ser errors en l'estil, zones de codi duplicat o no utilitzat, o parts de codi que poden ser propenses a errors, entre altres. Tots aquests problemes es mostren en una taula on apareix la quantitat de cadascun. Un altre lloc on els podem trobar és a la gràfica de qualitat, en la qual es mostra l'evolució de la qualitat del projecte. Aquesta s'obté de mesurar la quantitat d'errors en el codi, la complexitat d'aquest i la quantitat de codi repetit.

Acte seguit, podem veure el llistat de *commits* i *pull requests* que s'han anat realitzant juntament amb la quantitat d'errors que aquest ha generat i quins han siguts arreglats. També ens permet veure informació més detallada dels nostres fitxers mostrant-nos, per exemple, la seva complexitat i els errors que es troben en aquests. En el cas dels errors, aquests apareixen ressaltats sobre el codi, juntament amb el motiu i una breu explicació d'aquest, cosa que facilita molt la seva correcció.

Finalment, Codacy disposa d'integració amb Github, permetent així enllaçar el nostre repositori amb el revisor. D'aquesta manera, podem configurar que s'iniciïn revisions del codi cada cop que es realitzi un *commit* o un *pull request* a una determinada branca. A més, Codacy s'encarrega d'enviar els resultats de les revisions com a resposta de l'acció que l'ha activat. Això ens permetrà detectar i corregir ràpidament possibles problemes al nou codi, sense haver de perdre temps en que un dels revisors el llegeixi sencer.

7.9. Android studio

Per tal de desenvolupar el nostre projecte, utilitzarem una de les eines més completes que hi ha actualment per al desenvolupament d'aplicacions per a mòbils anomenada Android Studio. Aquest IDE especialitzat en aplicacions android, disposa d'un conjunt d'eines per a realitzar tant la interfície gràfica de l'aplicació com la lògica d'aquesta, de forma senzilla i fàcilment usable.

Per una banda, les aplicacions android utilitzen XML per tal de dissenyar l'aparença de les pantalles. Malgrat tot, aquest llenguatge pot ser difícil d'utilitzar, sobretot si no s'hi està acostumat. Per aquest motiu, aquest entorn de desenvolupament ofereix la possibilitat de dissenyar-les utilitzant un entorn gràfic, en el qual els components es poden col·locar arrossegant-los a la posició desitjada dins d'un contenidor d'elements i es poden modificar cadascuna de les seves propietats fàcilment.

Per l'altra banda, per tal de dissenyar la lògica de l'aplicació, aquest IDE disposa d'unes eines eficients per a mantenir el codi organitzat, llegible i eficient. A més, es poden afegir eines de testeig, com per exemple JUnit, de forma senzilla i s'ofereix una vista de testeig pròpia per aquestes.

Finalment, a part de poder desenvolupar l'aplicació, aquest entorn de desenvolupament es pot sincronitzar amb una eina de control de versions, com per exemple Github. Des del mateix programa, és possible crear noves branques, fer *commits* i *pushes* al repositori i iniciar els *pull request* per tal de realitzar la fusió entre les branques del projecte. D'aquesta manera no és necessari utilitzar un intèrpret de comandes per tal de realitzar aquestes accions en el repositori.

