Web App per monitoritzar un sistema de màquines remotes.

Arnau Sintes Rello

Resum— Aquest treball es basa en el desenvolupament d'una aplicació web que permeti controlar un conjunts de màquines clients de manera remota. L'aplicació web ha de mostrar la informació necessària i l'estat de cada màquina remota, dispositius connectats a les màquines i el software instal·lat.

Paraules clau— Monitoring, Web App, Host, Administradors, Web Service, Infraestructura IT, SpringBoot, VueJS.

Abstract—The objective of this project is the development of a web application that allow monitoring remote hosts. The web application must show the necessary information and the status of ecah remote Machine, devices Connected and the software installed.

Index Terms—Monitoring, Web App, Host, Administrators, Web Service, IT Infrastructure, SpringBoot, VueJS.

1 Introducció

AQUEST treball tracta el desenvolupament d'una aplicació web que permeti controlar un conjunt de màquines clients de manera remota.

Aquest monitoratge es basa en la detecció dels dispositius connectats i l'estat de cada un d'ells per tenir un control global del sistema. Les màquines remotes tenen dues tasques: validar una televisió digital i gravar streams de senyal de televisió digital. El servidor web ha de ser capaç de controlar l'estat el més "real-time" possible de tots els dispositius connectats a les màquines client.

El propòsit d'aquest document és donar una anàlisi detallada dels requisits de l'aplicació web a desenvolupar. Es descriurà l'arquitectura del programari i el disseny del model de dades en funció de l'anàlisi de requisits.

També podrem trobar en aquest document com s'ha realitzat el desenvolupament del projecte i la seva planificació.

2 Estat de l'Art

Actualment en el mercat hi ha diversos sistemes tant aplicacions d'escriptori com aplicacions web dedicades al monitoratge de sistemes remots. Una anàlisi de mercat ens serà útil per tal de definir les característiques bàsiques que ha de tenir la nostra solució i sobre tot trobar una manera eficient i còmode de mostrar tota la informació que tindrà la nostra aplicació web.

A continuació es mostra un llistat amb l'anàlisi de diferents aplicacions que hi ha al mercat, les quals són de les més conegudes. Es farà un petit resum, fent èmfasis en les característiques principals i fortaleses.

Nagios XI és una solució proposada per l'empresa Nagios que ens permet realitzar el monitoratge sobre aplicacions, servidors Windows o bé sobre l'estat de la xarxa. De fet Nagios XI va més enllà del monitoratge, permet una visió detallada de tots els components de la infraestructura i una detecció anticipada de problemes.

A continuació ressaltem els punts forts de Nagios XI:

- Permet controlar l'estat de cada host, els dispositius connectats, aplicacions instal·lades i els serveis.
- Interfície d'usuari simple i personalitzable.
- Servei de Reporting, que permet generar informes detallats de forma periòdica.
- Sistema de notificacions d'alertes. Permet Classificar i prioritzar les alertes segons els criteris dels usuaris (E-mail, mòbil, notificacions).
- Permet integrar sistemes de solucions/tiquets de tercers.

També cal remarcar que Nagios XI requereix que cada dispositiu que es vulgui monitorar, s'ha que configurar mitjançant un fitxer de configuració.

- *E-mail de contacte: arnau.sintes@e-campus.uab.cat*
- *Menció realitzada: Enginyeria del Software.*
- *Treball tutoritzat per: nom i cognoms del tutor (departament)*
- *Curs 2017/18*

Zabbix és una solució open-source que permet monitorar tot tipus de host, dispositius, serveis i aplicacions connectats en una xarxa.

Els punts forts que podem destacar són els següents:

- Control de tot tipus de sistemes, apps, serveis connectats a una xarxa.
- Permet monitorar webs i Servidors d'aplicacions Java.
- Facilitat d'escalabilitat, orientat per a tot tipus d'empreses i ecosistemes.
- Servei d'alertes personalitzable.
- Interfície web molt intuïtiva i fàcil, informació es mostra en gràfics interactius, és a dir Dashboards. També es poden generar mapes.
- Configurar solucions automàtiques mitjançant scripts.
- Monitoratge centralitzat que emmagatzema tota la informació en una Base de Dades relacional.

L'empresa **Groundwork** ofereix una solució open-source, anomenada "Monitor" que ens permet monitorar la xarxa, aplicacions, cloud i entorns hardware. Monitor ens ofereix

- Una interfície web simple.
- Estat dels sistemes a temps real.
- Un històric que permet fer anàlisi de tendències del sistema.
- També conte un llistat de possibles errades trobades.
- Ofereix un sistema de seguiment per la resolució de problemes.

Icinga2 és un sistema open source sota llicencia GNU2 que permet fer monitoring de la xarxa, dels sistemes desplegats en la infraestructura IT. Icinga és un fork de Nagios i permet la integració de tots els plugins de Nagios.
Les característiques que es poden aplicar al nostre sistema són:

- Ofereix dos tipus d'interfícies.
- Visualització de la informació mitjançant dashboards.
- Generació d'un històric dels estats dels host, notificacions, o errades.
- Notificacions als usuaris.
- Definició de diferents nivells d'alertes.
- Genera reports de rendiments.
- Conté una API REST que permet una fàcil integració dels nous mòduls.

Ansible tower és una eina d'automatització IT que permet a més el control de la infraestructura IT. Ansible tower compta amb una API REST i un CLIENT que permet integrar-se a les eines existents.
Com a funcionalitats interessants i aplicables al nostre sistema té:

- Interfície d'usuari simple i intuïtiva.
- Mostra la informació de tota la infraestructura IT mitjançant "dashbboards".
- Notificacions integrades del sistema.
- Fàcilment escalable, no hi ha límit de nodes connectats.
- Administració d'usuaris i rols.
- Ofereix un inventari de tots els host i dispositius o sistemes de la infraestructura IT.

Pel que fa a Ansible AWX, és el mateix projecte que Ansible Tower però open-source sota la llicència d'Apache versió 2.0. La diferència entre els dos projectes és que Ansible Tower ofereix un servei de manteniment i actualitzacions i suport per incidències.

Com hem pogut observar al mercat existeixen moltes solucions a escollir, moltes d'elles són open-source, l'afavoreix a les empreses a no haver de realitzar una important inversió. A més totes permeten integració de plugins o paquets per millorar les seves funcionalitats.

Com a resum de l'estat de l'Art veurem una taula comparativa amb les característiques principals que volem que tingui la nostra aplicació web i veure quines aplicacions del mercat les satisfan:

	Temps real	*Notificacions*	*Històric d'alarmes*	*Troubleshooting editable*
Nagios	Si	Si	Si	No
Zabbix	Si	Si	Si	No
Groundwork	Si	Si	Si	No
Icinga2	Si	Si	Si	No
Ansible Tower	Si	Si	Si	No

Taula 1 - Estat de l'Art

3 OBJECTIUS

El sistema proposat té com a propòsit principal controlar la salut d'una infraestructura IT concreta. A més de la creació d'alarmes i notificacions en cas de fallida. Un dels propòsits principals del projecte, que marca la diferència respecte a altres solucions del mercat, és la integració d'un manual de solucions. En l'apartat 4 Descripció General, trobarem una descripció detallada de les característiques i estructura del sistema a desenvolupar.

El projecte té com a objectiu principal posar en pràctica tots els coneixements obtinguts en el Grau d'enginyeria Informàtica, a més d'aprofundir en aplicar tècniques de treball i desenvolupament que estan a la vanguardia en el món empresarial, i per últim la introducció de noves tecnologies que poden aportar un valor afegit al projecte i permeten a l'estudiant enriquir els seus coneixements i habilitats.

4 DESCRIPCIÓ GENERAL

Es tracta d'un sistema que permetrà als administradors tenir un control de tots els hosts que hi ha dispersos pels diferents països. Els permetrà també, saber en quin estat es troben els components que té connectats els host i les aplicacions instal·lades.

El sistema consta de:

- Aplicació web i una base de dades desplegats en un servidor central.
- Hosts connectats a la mateixa xarxa que el servidor central. Cada host consta de diferents dispositius connectats i d'aplicacions instal·lades.
- Una aplicació instal·lada a cada host que es comprova l'estat del host quan l'aplicació web li sol·licita.

Les funcionalitats principals que aporta aquest sistema a més del monitoratge de hosts de forma general o detallada d'un únic host són:

- Detecció de dispositius connectats per USB.
- Anàlisi de l'estat d'aplicacions i serveis del sistema.
- Arrancada i aturada d'aplicacions i serveis.
- Anàlisi de l'estat dels discs durs del sistema i els discs durs externs connectats.
- Administració d'aplicacions, serveis, dispositius i discs durs a monitorar.
- Manual de solucions.
- Administració del manual de solucions.
- Generació d'alarmes.
- Possibilitat d'enviar alarmes via correu electrònic.
- Inclou dos modes de funcionament: Sota demanda o actiu.

El sistema pot ser executat de forma activa, és a dir, el sistema de forma automàtica, monitoritza constantment la infraestructura IT. O bé, pot ser executat sota demanda, de manera que a petició dels usuaris comprova l'estat de la infraestructura. Aquest últim permet no sobre carregar de peticions els hosts i pot ser útils per a infraestructures senzilles i poc potents.

4.1 Arquitectura del Sistema

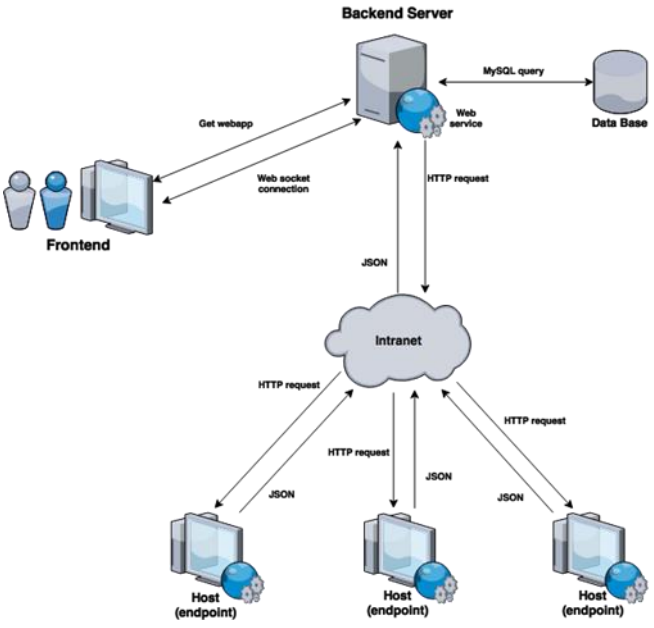


Figura 1 - Arquitectura de Sistema

4.2 Arquitectura del Software

En el disseny de l'arquitectura del software d'aquesta aplicació es pot veure una clara separació entre el concepte model i vista.

El model emmagatzema les dades i també conté funcionalitats de l'aplicació mentre la vista és la representació visual del model.
Aquesta separació permet reduir la dependència entre el comportament de l'aplicació i la forma en què es representa el model l'usuari.

A més aporta avantatges molt importants com la separació de responsabilitats que ens permetria accedir al model amb diferents clients, ja que no hi ha dependència del model amb la vista. Ens facilita manteniment, en poder actualitzar o fer refactor de la vista sense fer grans canvis al model.

Per últim destacar, que permet el desenvolupament en paral·lel per diferents equips

Com es pot observar al Diagrama de Components que es pot trobar a l'apèndix A.1 Diagrama de Components, l'arquitectura del sistema d'aquesta aplicació està compost per 4 components principals: la vista, el back-end, la base de dades i una aplicació arrencada a cada host que comprova l'estat del host. A més en el diagrama es pot veure la interacció i estructura dels hosts a monitorar.

5 ANÀLISI DE REQUISITS

Per l'anàlisi de requisits s'han dut a terme diferent tècniques de suport a l'elicitació de requisits, les quals es poden veure detalladament al llarg d'aquest punt. El principal punt d'entrada de requisits han estat les reunions inicials amb el client i possibles usuaris del sistema. Tots aquests requisits han estat recollits i classificats segons les seves característiques.

Es va realitzar una primera fase d'anàlisi de requisits, on es van recollir la gran majoria d'aquests, que posteriorment van ser acceptats per ambdues parts implicades al projecte, client i equip de desenvolupament, i no han rebut gaires canvis.

Per altra banda, sí que es van realitzar alguns canvis a les restriccions tècniques projectes, és a dir aquells requisits que impliquen metodologies o tecnologies a aplicar en el sistema. Concretament han estat les tecnologies del front-tend i Middelware.

S'ha desenvolupat fen servir Vue JS. Finalment s'ha escollit aquest framework amb l'objectiu de donar frescor i generar interès en tractar-se d'una tecnologia relativament nova i amb un gran suport per la comunitat de programadors. Tot i ser una tecnologia nova per a l'equip de desenvolupadors, hem optat per ella, ja que facilita la feina a l'hora de desenvolupar i el cost que suposa aprendre no té un gran impacte a la fase de desenvolupament.

Per últim el protocol de comunicació entre la vista i el model de l'aplicació ha estat modificat per una API REST degut al canvi de tecnologia de la interfície d'usuari. La gestió de web Sockets amb ReactJS, que era la proposta inicial com a tecnologia del front-end, és molt eficient i relativament senzilla d'implementar, en canvi amb les noves tecnologies no és ni eficient ni senzilla.

5.1 Requisits Funcionals

ID	*Descripció*
RF001	Controlar l'estat dels diferents hosts des d'una web app.
RF002	L'aplicació web ha de tindre una wiki que doni informació sobre com solucionar alguns errors.(Troubleshooting)
RF003	L'aplicació ha de permetre modificar la wiki de solució de problemes (Troubleshooting).
RF004	En cas d'error o estat erroni s'han de proporcionar accés ràpid a possibles accions per resoldre (troubleshooting).
RF005	El sistema ha de mostrar l'estat general de tots els hosts.
RF006	L'aplicació ha de mostrar un resum de l'estat de la infraestructura i una gràfica de tendències.
RF007	L'aplicació web ha de mostrar l'estat dels serveis crítics i un històric de les darreres alertes del host
RF008	El sistema ha de generar alarmes per a cada canvi d'estat dels diferents sistemes disponibles en el host.
RF009	L'administrador, ha de poder accedir a la informació detallada de cada host
RF010	L'aplicació web ha de mostrar la informació a temps real.
RF011	L'aplicació web ha de mostrar quan s'ha actualitzat l'estat d'un host i els seus components.

Taula 2 - Requisits Funcionals

5.2 Requisits No Funcionals

ID	*Descripció*
RNF001	El sistema ha de ser capaç de proporcionar informació de l'estat dels host de la forma més real-time possible.

Taula 3 - Requisits No Funcionals

5.3 Restriccions

ID	*Descripció*
RS001	El sistema de Monitor és una Web App.
RS002	El sistema tindrà un WebService amb tecnologia REST.
RS003	El sistema comptarà amb la tecnologia de API Rest com mètode de connexió entre clients i servidor.
RS004	El back-end del sistema es realitzarà amb Java.
RS005	El sistema tindrà una base de dades MySQL.
RS006	El sistema s'ha de desenvolupar fent servir Agile Development.
RS007	La interfície d'usuari es realitzarà emprant el framework Vue JS.

Taula 4 - Restriccions

5.4 User Journey Map

Per completar l'anàlisi de requisits s'ha realitzat un "User Journey Map". Es tracta d'una tècnica que ens permet definir l'experiència que té un usuari en fer servir un producte o servei, en el nostre cas, és l'experiència que tindrà un administrador en fer servir la nostra aplicació web. En un "User Journey Map" es defineixen les fases i activitats que desenvoluparà un usuari en la nostra aplicació com també els canals d'interacció i les emocions que pot tenir durant les diferents fases d'interacció.

5.5 Story Mapping

Per requeriments del client s'ha realitzat un Story Mapping, Figura 2 – Story Mapping, el qual és una altra tècnica de suport d'elicitació de requeriments que s'ha realitzat a partir del User Journey Map , anteriorment esmentat. Aquesta tècnica ens permet, a partir d'un determinat Stakeholder, recollir una visió global del projecte però també el detall de les funcionalitats requerides.

Es tracta d'una tècnica molt útil tant pel client com per l'equip de desenvolupament, ja que el fet d'agrupar cada

funcionalitat i tasques principals en base un objectius permet definir l'espina dorsal del projecte i facilita la planificació de les tasques.

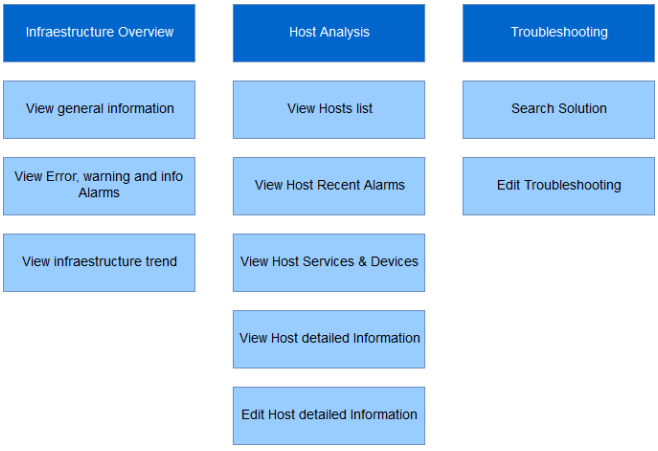


Figura 2 - Story Mapping

6 DESENVOLUPAMENT

6.1 Històries d'usuari

Prèviament a començar la programació del sistema, s'havien de definir les tasques a seguir, per això s'han creat les històries d'usuari.

Les històries d'usuari s'empren com a objectius o guia a seguir, durant el desenvolupament del sistema, ja que a partir d'aquestes es realitza una descomposició de tasques per tal d'assolir un objectiu, el qual està definit a les històries d'usuari.

ID	*Descripció*
US001	Com a administrador, vull saber l'estat de general de la infraestructura IT.
US002	Com a administrador, vull tenir un llistat de tots els hosts amb informació del seu estat.
US003	Com a administrador, vull tenir un llistat d'alarmes de cada host.
US004	Com a administrador, vull tenir un llistat amb l'estat dels serveis i dispositius de cada host.
US005	Com a administrador, vull veure la informació detallada d'un host.
US006	Com a administrador, vull editar la informació detallada d'un host.
US007	Com a administrador, vull tenir un manual de solucions a la web.
US008	Com a administrador, vull editar les entrades del manual de solucions.

Taula 5 - Històries d'Usuari

6.2 Mockups

A partir de les Històries d'usuari es va fer una representació en mockups del sistema final. Els quals no representen una versió definitiva de l'aspecte de l'aplicació web. Si no, que són una representació gràfica de les funcionalitats descrites anteriorment.

Els mockups permeten al client tindre una visió més detallada del sistema, tenint en compte que no són una còpia exacta de l'aspecte final de l'aplicació. A més, és una guia per als desenvolupadors alhora de desenvolupar tant la interfície gràfica com el model de l'aplicació.

A l'apèndix A.2 Mockups, es poden trobar alguns exemples dels mockups que s'han desenvolupat, no s'han adjuntat tots, ja que són moltes imatges, s'han escollit aquelles que representen els requisits principals.

6.3 Disseny de la Base de Dades

Un cop finalitzades les tasques d'anàlisi de requisits i els processos de disseny de les interfícies gràfiques, procedim a començar el desenvolupament de l'aplicació web. Primer de tot, necessitem una base de dades sobre la qual treballar.

S'han proposat diferents dissenys, i sobre el que s'està treballant i desenvolupant l'aplicació, comprèn totes les necessitats del sistema a l'hora de desenvolupar l'aplicació. Aquest disseny conté totes les entitats definides en el Model de domini, Figura 3 – Model de Domini, que representa les entitats del domini i com estan relacionades a escala conceptual.

A l'apèndix A.3 Diagrama de classes de la Base de Dades, podem trobar el Diagrama de classes de la base de dades.

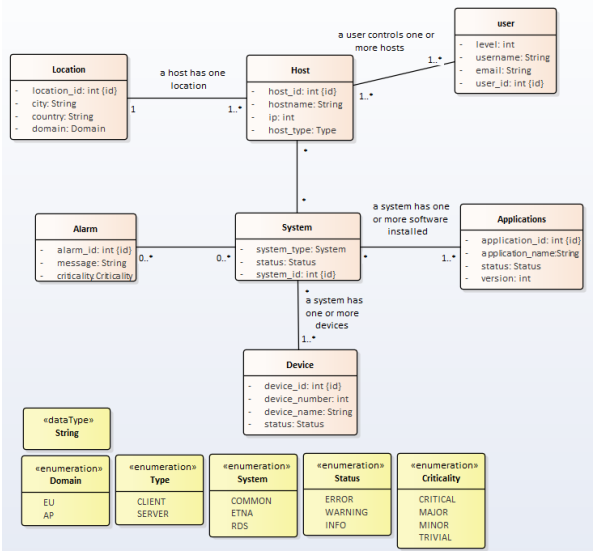


Figura 3 - Model de Domini

6.4 Desenvolupament del back-end.

Havent definit i creat totes les bases necessàries per a obtenir un desenvolupament encertat i sòlid, necessitem configurar un sistema de control de versions. Git ha estat l'escollit, ja que és, el que he après a fer servir a la universitat i el que es fa servir de forma corporativa a l'empresa.

El back-end de l'aplicació s'ha desenvolupat sota el Framework Spring Boot, es tracta d'un framework per a la comunitat Java basat en la convenció en comptes de la configuració, això vol dir essencialment, que el desenvolupador no ha de configurar tots els aspectes de l'aplicació sinó aquells que no són comuns. Spring Boot detecta certs elements del codi gràcies la convenció de noms. Un altre benefici que aporta Spring Boot i que és essencial per aquest sistema és la integració d'un servidor Tomcat, de manera que no s'han de generar fitxer ".war" i posteriorment desplegar-los en un tomcat al servidor central. Es genera un executable que només ha de ser executat al servidor central.

Per al desenvolupament del sistema es crearà una arquitectura modular basat en capes que permetin al sistema obtenir un baix índex d'acoblament entre mòduls i un elevat índex de cohesió entre ells. Es seguirà una estructura per capes, que és l'organització del projecte en quatre capes principals: presentació, aplicació, domini i persistència o infraestructura. Cada una de les capes té una funció determinada:

- **Presentació**: conté totes les classes encarregades de mostrar la vista al client final o bé la comunicació amb el back-end de l'aplicació.
- **Aplicació**: té tota la lògica de negoci que necessita l'aplicació per satisfer els requeriments. Bàsicament consisteix en serveis que interactues amb els objectes de la capa de domini per satisfer les històries d'usuari i casos d'ús definits.
- **Domini**: representa el domini de l'aplicació i està format per entitats.
- **Infraestructura**: conté totes les classes responsables de la persistència de les dades de la base de dades.

Un cop definida l'estructura de l'aplicació, s'ha realitzat una descomposició de tasques a seguir. Aquesta descomposició es pot veure a la Taula 4 – Tasques, que defineix l'ordre lògic a seguir per tal de crear satisfactòriament l'estructura anteriorment definida.

Tasca	*Descripció*
1	Configuració del DataSource.
2	Crear les entitats del model de domini.
3	Crear els repositoris o els DAO.
4	Creació de les interfícies dels serveis i les seves implementacions.
6	Creació dels DTO.
7	Definició de l'API Rest.
8	Creació dels Controlladors Rest

Taula 6 - Tasques

Tant els DAO com els DTO, són patrons de disseny. El primer, Data Acces Object, són objectes que proporcionen accés a una base de dades o altre tipus d'emmagatzematge persistent. Els DAO contenen tots els mètodes CRUD necessàries, normalment es té un DAO per a cada entitat que hi ha a l'aplicació. El segon, Data Transfer Object, és un patró que s'utilitza per transferir valors entre la vista i el servidor, permet aïllar la capa de la vista de la del domini.

A continuació podem veure en detall com és la comunicació entre les diferents capes de l'aplicació:

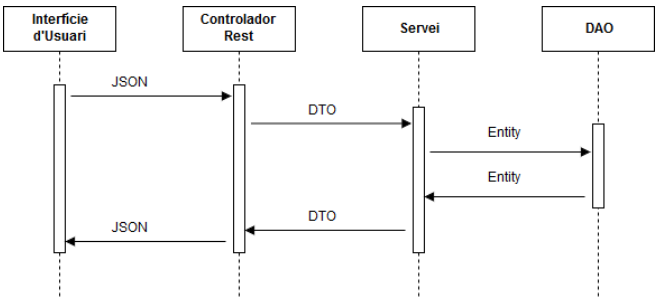


Figura 4 - Diagrama de Comunicació.

Per tal de realitzar l'anàlisi de la salut dels hosts, és necessària la implementació d'una aplicació node que està instal·lada a cada un dels hosts o màquines client.
La nova aplicació, node, s'ha desenvolupat també sota el framework Spring Boot, seguint la mateixa arquitectura per modular que el back-end, però en aquest cas no han estat necessàries les capes de domini ni persistència de dades.
La funcionalitat principal d'aquest node és respondre a les peticions rebudes del back-end, de manera que via API Rest és capaç de proporcionar informació al back-end sobre l'estat dels dispositius usb connectats, aplicacions i serveis que conté el host o bé d'informar sobre l'estat dels discs durs de la màquina. A més de poder arrencar i aturar serveis i aplicacions del host de forma remota.

6.5 Desenvolupament del Front-end.

La interfície d'usuari ha estat desenvolupada sota el Framework Vue JS, es tracta d'un framework per al desenvolupament d'aplicacions, especialment centrat en la vista i la interacció amb l'usuari. Com el seu nom indica està basat en Javascript i va ser publicat en 2014.

Aquest framework és famós per ser molt semblant a React JS i a Angular, les principals característiques que li atorguen el reconeixement de la comunitat són la implementació de DOM virtual, cosa que redueix el principal coll d'ampolla de les interfícies d'usuari, integració de components reutilitzables, models de vista basats en Html, o reactivitat la qual permet actualitzar la vista de forma automàtica en el cas que les variables JavaScript siguin modificades. A part de les característiques esmentades, uns dels principals motius per la que s'ha escollit aquest framework és per l'encaminament entre vistes i pàgines que ofereix facilitant molt la feina al desenvolupador.

Per al desenvolupament de la interfície d'usuari s'ha fet servir una plantilla, en concret la de Core UI, el qual està disponible en versió gratuïta sota llicència MIT o bé en la seva versió de pagament. La primera ha estat l'escollida, ja que incorpora molts components predefinits que faciliten el desenvolupament d'elements com taules, gràfiques, o formularis. A més incorpora per defecte molts mòduls que

faciliten la integració de noves funcionalitats al front-end.

El front-end és l'encarregat de mostrar visualment les dades proporcionades pel back-end. Com s'ha explicat anteriorment la comunicació és via API Rest, de manera que la vista fa peticions Rest per tal d'obtenir les dades, realitzar insercions de noves dades, edició o eliminació de les dades existents.

Mitjançant aquesta interfície, l'usuari pot realitzar algunes funcions a part de la principal que és el monitoratge dels hosts de la infraestructura IT. L'usuari pot donar d'alta per a un host noves aplicacions, serveis o especificacions tècniques d'aquest, a més de poder-les eliminar o editar. També té la possibilitat de consultar o editar un manual de solucions.

L'usuari pot realitzar diferents accions com refrescar l'estat d'un host o bé més d'un mitjançant un botó o bé arrencar i parar aplicacions o serveis per tal de monitoritzar els diferents hosts i administrar-los.

6.6 Tests

Durant el desenvolupament de l'aplicació s'ha dut a terme un conjunt de tècniques de test. En finalitzar el desenvolupament de noves funcionalitats es realitzaven tests d'integració i funcional, un cop construïda una versió estable de l'aplicació s'han realitzat tests funcionals en un entorn de pre-producció, el qual simula de la manera més acurada possible, l'entorn on treballarà l'aplicació un cop entregada al client.

Els tests funcionals s'han dut a terme amb diferents usuaris de les o perfils de treballadors susceptibles a fer servir aquesta eina, es tracta d'una tècnica anomenada entrega de canaris o d'amics, on s'escull un nombre de persones dins de l'entorn de coneguts per tal de provar l'estat de l'aplicació i obtenir feedback de les seves opinions. L'obtenció de feedback ha estat mitjançant reunions on es recollien aspectes positius, negatius i possibles millores o funcionalitats que podria obtenir l'aplicació. Totes aquestes reunions han estat documentades per tal de tenir un registre de la feina realitzada.

7 METODOLOGIA

La metodologia escollida per al desenvolupament de l'aplicació ha estat SCRUM.

Es tracta d'una metodologia de desenvolupament Agile que com a principal objectiu té la minimització dels riscos, ja que, està basada en una planificació iterativa i no lineal com pot ser una metodologia en cascada.

Els principals beneficis que obtenim a l'hora de treballar amb una metodologia Agile és la millora de reacció enfront de canvis, és a dir, ens aporta una agilitat a reaccionar a canvis que sorgeixen durant el desenvolupament del projecte.

En SCRUM es realitzen diferents iteracions, anomenades Sprints, on es duen a terme cada una les fases d'un projecte: planificació, desenvolupament, revisió i obtenció de feedback o retrospectiva.

Figura 5 - Scrum Cycle

Per al desenvolupament de l'aplicació web s'han definit sprints d'entre 1 a 2 setmanes, per a poder tenir un control i un feedback constant del client, cosa que ens permetrà centrar-nos en el que realment vol.

La forma de treballar que s'ha aplicat, és la pròpia d'un projecte Agile, s'han definit mitjançant històries d'usuari totes les funcionalitats que han estat extretes del procés de captura de requisits i han estat agrupades en el Produt Backlog.

A l'inici de cada Sprint s'ha realitzat una descomposició en tasques de cada una de les històries d'usuari, escollides per al Sprint, i finalment es valorava cada una en funció de la dedicació que requeria. Aquestes reunions anomenades Sprint Planning, s'han fet conjuntament amb el client, de manera que es té una visibilitat i una transparència, tant per part de l'equip en saber que és el que vol el client, com del mateix client en saber que està fent l'equip de desenvolupament, durant tot el procés de desenvolupament.

En finalitzar un Sprint, es realitza una reunió de demostació on s'entrega un MVP, o Minimal Viable Product, el qual es tracta d'una entrega o bé un artefacte nou que aporta valor al client i qualitat d'una manera acurada, és a dir no es realitza més treball del que espera el client.

Com em pogut veure es tracta d'una metodologia centrada al client i en la satisfacció d'aquest, donant molt èmfasi en la transparència entre el client i l'equip de desenvolupament i la reducció de risc.

8 PLANIFICACIÓ.

Tot i treballar sota una metodologia Agile es poden identificar 4 fases clarament: inici, planificació, desenvolupament i entrega. A cada una de les fases s'apliquen les tècniques i bones pràctiques d'Scrum per a duur a terme l'objectiu global de cada fase, cal destacar que a la fase de desenvolupament s'han fet els Sprints d'una forma més teòrica i estricte que no pas a les altres etapes del projecte.
A continuació veurem una descomposició de les tasques a realitzar

Fase	*Tasca*
Inici	Reunió amb el tutor

	Reunió amb el client
	Reunió amb el equip
Planificació	Anàlisi de recursos
	Anàlisis de eines
	Definició de tasques
	Anàlisis de costos
	Anàlisis de requisits
	Lliurament Informe Inicial
Desenvolupament	Sprint 1
	Sprint 2
	Sprint 3
	Lliurament Informe de Progrés 1
	Sprint 4
	Sprint 5
	Documentació Sprints 4 i 5
	Lliurament Informe de Progrés 2
	Sprint 6
	Documentació Sprints 6
	Proposta de Informe Final
Entrega	Documentació final del projecte
	Proposta de Presentació
	Lliurament final del projecte
	Preparació de la defensa del TFG
	Defensa del TFG

Taula 7 - Descomposició de Tasques

8.1 Desviacions

Durant l'execució de la fase de desenvolupament ens hem trobat amb nous requisits per part del client o bé tasques que han suposat una dedicació més gran que la inicialment estimada. Concretament al Sprint 1, es van realitzar tasques d'anàlisi de requisits a petició del client, cosa que va suposar dedicar temps inicialment planificat per a desenvolupament. Aquesta petició va tenir un impacte baix en el sprint però no en l'objectiu principal d'aquest. A la Taula 8 – Durada de Tasques, podem veure com ha estat aquest impacte.

Al Sprint 4, concretament en el desenvolupament de l'aplicació node es va trobar una tasca bloquejant, ja que afectava directament un requisit funcional, ha estat l'anàlisi de dispositius usb. El fet de ser una aplicació Java per a sistemes operatius Windows limitava molt les possibilitats de detecció de dispositius USB, una primera solució va ser l'ús de la famosa llibreria libusb, realitzada en C, amb un wrapper per a poder ser cridada en Java, tot i això no era la solució òptima de manera que es va seguir investigant per tal d'implementar una solució adequada, la qual va ser l'ús de la llibreria OSHI, Operating System and Hardware Information, disponible a Github sota llicència EPL 1.0 compatible amb projectes Open Source i comercials.

El desenvolupament de la detecció de dispositius usb va suposar molta més feina de la inicialment esperada, de manera que la meitat del Sprint 4 va ser dedicat a aquesta tasca, la resta del sprint es va finalitzar el desenvolupament de la proposta inicial de l'aplicació node. Com podem veure a la Taula 9 – Descomposició Sprint 4 i 5 aquesta tasca bloquejant va suposar un canvi en la planificació del Sprint 4, on inicialment a la primera setmana del Sprint 4 es realitzaria el node i la segona setmana es realitzaria la interfície d'usuari. De manera que tot l'Sprint 4 es va dedicar a la solució de la detecció de dispositius usb i el Sprint 5 al desenvolupament del front-end.

Tasca	*Planificació Inicial*	*Nova Planificació*
Anàlisis de requisits. Sprint 1	27 hores 40 hores	39 hores 28 hores

Taula 8 – Durada de Tasques

Sprint	**Tasca**	Planificació Inicial	Nova Planificació
4	*Desenvolupament Node*	*20 hores*	*40 hores*
	Desenvolupament Front-end	*20 hores*	*0 hores*
5	*Desenvolupament Front-end*	*40 hores*	*40 hores*

Taula 9 – Descomposició Sprint 4 i 5

Tot i incorporar noves tasques als Sprints o trobar-ne de bloquejant no ha suposat un impacte en la planificació del projecte. Principalment és gràcies a la metodologia Agile seguida i una de les seves bones pràctiques a l'hora de planificar un Sprint. No s'ha de planificar el 100% de carrega de treball assumible en un Sprint sinó que s'ha de planificar entorn del 80% per tal de poder fer front a imprevistos i incidències. D'aquesta manera s'ha pogut respondre de forma ràpida i eficaç a tots els possibles inconvenients trobats durant de l'execució del projecte.

9 Costos

En aquest apartat es descriuran i es farà un anàlisi dels costos associats al desenvolupament de l'aplicació web. Finalment, es farà una esmena del possible impacte que han tingut les desviacions produïdes el desenvolupament del projecte.

Primer de tot, hem de tenir en compte els costos indirectes de material, eines i llicències que han estat assumits. Quan a costos de material només tenim l'ordinador portàtil que facilita l'empresa per al desenvolupament del projecte. El qual està valorat en 650 €.

Eina	***Tipus Llicència***	***Cost***
Eclipse Oxygen	EPL	0€
Insomnia	MIT	0€
Sublime Text	Proba	0€
GSuite	Enterprise Edition	25€/mes
Enterprise Architect.	Software Engineering Edition	400€
Draw.io	EULA	0€
JIRA	Premiun	5€/mes
BitBucket	Premium	7€/mes
MySQL Server and Workbrench	GPL	0€
Plantilla Core UI	MIT	0€

Taula 10′ - Costos Indirectes

El total dels costos indirectes ascendeix a 540, com es pot observar en la Taula 10 Costos Indirectes. De manera que els costos indirectes augmenten fins a uns 1.190 €.

En tractar-se d'un projecte intern de l'empresa, el projecte ha estat desenvolupat com a treballadors, per tant, no s'analitzen ni es desglossen les hores dels diferents perfils que han intervingut en el projecte. Si no, que hem d'analitzar la viabilitat del projecte i que pot facilitar el treball als futurs usuaris de l'aplicació. Primer de tot hem de tenir en compte que l'eina buscar solucionar els problemes que els testers es troben quan volen realitzar un test manual o en programen un d'automàtic i en ser executat falla, ja que algun dispositiu o software no està connectat o arrencat. De mitja es triguen entre 8 hores a 1 dia en resoldre aquestes incidències, ja que els hosts estan distribuïts arreu del món i la política de solucions és la de contactar amb la persona encarregada del sistema i verificar l'estat i solucionar. Per tant aquesta eina permet identificar l'estat del sistema de forma prèvia i en cas d'haver-hi error, identificar-lo i saber que és el que passa concretament. De manera que permet una acció indirecta en cas de ser una errada de software o serveis reduint l'espera de 8 hores o 1 dia fins a 1 hora a molt tardar. També permet que el suport per tal de la persona encarregada del sistema en cas d'errada harware disminueix fins a la meitat.

Com hem pogut veure es redueix els temps d'espera en més del 50%, provocant que el rendiment dels testers augmenti. De manera que el cost del projecte és assumible per part de l'empresa, tot i significar una inversió inicial, a la llarga els aporta beneficis indirectes que justifiquen el desenvolupament de l'aplicació.

En la fase de desenvolupament s'han produït alguns canvis i inconvenients que ha pogut afectar la planificació inicial, però no als temps ni milestones definits, de manera que no ha suposat cap increment en els costos. El cavi que podria haver suposat un impacte més brusc a la panificació a estat el canvi de tecnologia pel front-end, però gràcies a l'experiència amb tecnologies similars i el suport de companys, s'ha minimitzat molt l'impacte inicial que podria suposar tant en la planificació com a costos. A més, en poder anar sobre planificat, ens permet fer testing de l'aplicació, cosa que pot permetre reduir l'esforç a dedicar al projecte en un futur per arreglar possibles errors, cosa que implicaria un augment del cost del projecte.

10 RESULTATS

Tot i existir desviacions durant la fase de desenvolupament, al final del projecte s'ha pogut entregar un producte totalment funcional que compleix tant amb els requisits funcionals inicialment definits com amb els no funcionals. Posteriorment a la fase de test, el feedback obtingut dels usuaris ha estat molt positiu, ja que trobàvem l'eina fàcil de fer servir i útil, de manera que els aportava valor a la seva tasca diària. A l'apèndix A.4 Resultats, podem veure algunes captures de pantalla de la interfície web de l'aplicació on es poden observar totes les funcionalitats desenvolupades.

11 MILLORES FUTURES

Un dels principals punt de millora que té l'aplicació és el sistema Node, que com hem dit abans, s'encarrega d'interactuar amb el host a monitorar. La idea seria substituir aquest Node per una solució Open Source com Ansible AWX, la qual té moltes funcionalitats interessants que el nostre node no té, i les podríem aprofitar per a desenvolupar funcionalitats que puguin ser atractives i útils per als usuaris. La seva integració seria relativament fàcil i aportaria molt valor al client, ja que se li ofereix una solució contrastada en el mercat. A més facilita gestionar tot tipus de sistemes operatius, ja que el nostre sistema està limitat actualment al sistema operatiu Windows.

Prèviament a la posada en producció del sistema cal implementar un sistema d'autenticació segur, sent una de les opcions l'autenticació a través de LDAPS.

A més de millores en el procés de desenvolupament i entregues, com pot ser la introducció de conceptes DevOps, com unit testing, integració continua i entrega automàtica. Aquests aspectes aporten beneficis a l'equip de desenvolupament i al client, facilita tasques rutinàries i ajuda a millorar l'estabilitat de l'aplicació.

12 CONCLUSIÓ

Per poder desenvolupar un nou projecte de forma estable, és necessàri tenir una bona base sobre la que començar. Aquesta base ens l'ofereix un bon anàlisi de requisits, que en primer lloc ens permet saber que és realment el que vol el client, a més millora la capacitat de planificació del projecte, disminuint les despeses i possibles endarreriments en el projecte. Tot això implica una millora de la qualitat del producte resultant.

El fet de fer ús d'eines i entorns de treballs innovadors o nous, ens aporta una idea de com es treballa actualment, a la vegada que ens ofereix les facilitats de les noves tecnologies com pot ser el framework Spring Boot, el qual ens facilita molt la feina de tenir una aplicació web funcional sense requerir molt esforç en la configuració.

Haven fet un bòn anàlisi de requisits i obtenir un sistema estable no garanteix l'èxit. Com a bons enginyers de software hem de desenvolupar software intuïtiu i durader, aquesta característica ve donada per la mantenibilitat del sistema. En la nostra WebApp el disseny modular ens garanteix un molt bon índex de mantenibilitat del software, ja que tots tres mòduls, front-end, back-end i node, són independents entre ells. També facilita el desenvolupament entre equips, ja que no existeixen dependències entre

projectes. Per poder aprofitar aquest benefici és molt important documentar de forma efectiva la comunicació que hi ha entre els mòduls, és a dir, quines dades es comparteixen entre ells.

Tot i tenir bones eines i entorns de treballs, grans anàlisis de requisits o desenvolupar software de qualitat, el que marca la satisfacció d'un client enfront un nou projecte és una bona planificació, si aquesta se segueix de manera correcta, si no hi ha desviacions o inconvenients o bé si el producte final és el pactat. És important que en trobar alguna tasca bloquejant es dediqui temps a resoldre-la però no ens hem d'oblidar de l'objectiu global de la tasca, ja que podríem perdre la visió global del projecte. També és molt important en aquests casos, saber prioritzar entre tasques o objectius del projecte i per últim, destacar la importància de saber demanar ajuda i no prendre's les tasques bloquejants com a personals, deixar-se aconsellar per companys amb més experiència, aquesta actitud ens permet obtenir una altra visió sobre els problemes que ens anem trobant i poden facilitar la resolució d'aquests.

Per últim destacar que dur a terme el projecte des de la seva captura de requisits fins a la seva entrega, m'ha permès tenir una visió completa de l'envergadura d'un projecte i que és el que implica. Ha estat una bona oportunitat per arriscar a fer servir només eines i tecnologies que en altra situació no hagués fet servir, i m'han proporcionat nous coneixements que poden ser útils per un futur.

AGRÏMENTS

Els agraïments del projecte van dirigits en primer lloc als dos tutors, Marc Talló i Maria Dolors Mimó, que m'han ajudat a enfocar el projecte i guiat durant la durada d'aquest per tal d'aconseguir l'objectiu principal.
Agrair a l'empresa IDNEO Technologies SLU, per oferirme l'oportunitat de desenvolupar el Treball de Fi de Grau en un entorn de treball molt professional i amb grans persones que no han dubtat en donar-me suport i m'han permès créixer com a futur enginyer.

BIBLIOGRAFIA

[1] Nagios XI [Consulta Febrer 2018] Disponible a: https://www.nagios.com/products/nagios-xi/.
[2] Zabbix [Consulta Febrer 2018] Disponible a: https://www.zabbix.com/.
[3] Groundwork [Consulta Març 2018] Disponible a: http://www.gwos.com/.
[4] Icinga2 [Consulta Març 2018] Disponible a: https://www.icinga.com/products/icinga-2/.
[5] Ansible [Consulta Març 2018] Disponible a: https://www.ansible.com/products/tower.
[6] Scrum [Consulta ta Març 2018] Disponible a: https://www.scrum.org/resources/what-is-scrum.
[7] Elicitació de Requisits [Consulta Març 2018] Disponible a: https://e-aules.uab.cat/2017-18/mod/folder/view.php?id=40691.
[8] Documentació Spring Boot [Consulta Març 2018] Disponible a: https://docs.spring.io/spring-boot/docs/current-SNAPSHOT/reference/htmlsingle/ .
[9] Guies de Spring Boot [Consulta Març 2018] Disponible a: https://spring.io/guides
[10] Blog de tutorials de desenvolupament Java web [Consulta Abril 2018] Disponible a:http://www.mkyong.com/tutorials/spring-boot-tutorials/.
[11] Tutorial i exemples de Spring Boot web development [Consulta Abri 2018] Disponible a: http://www.baeldung.com/ .
[12] Documentació JPA Repositories de Spring [Consulta Abril 2018] Disponible a: https://docs.spring.io/spring-data/jpa/docs/1.5.0.RC1/reference/html/jpa.repositories.html .
[13] Fòrum de programació per a resolució de preguntes. [Consulta Abril 2018] Disponible a: https://stackoverflow.com/ .
[14] Documentació Apache Maven Project [Consulta Abril 2018] Disponible a: http://maven.apache.org/guides/ .
[15] Suport a la Elicitació de Requisits [Consulta Març 2018] Disponible a: https://e-aules.uab.cat/2017-18/mod/folder/view.php?id=40691.
[16] Pàgina oficial de Vue JS [Consulta Maig 2018] Disponible a: https://vuejs.org/
[17] Documentació de Bootstrap per a Vue JS [Consulta Maig 2018] Disponible a: https://bootstrap-vue.js.org/

GLOSSARI

Host: Qualsevol ordinador o altre dispositiu connectat a una xarxa que proporcionen o fan servir serveis de la xarxa.
Infraestructura IT: conjunt de dispositius i aplicacions connectats que donen un servei a una empresa.
Troubleshooting Manual: Manual que conté possibles solucions per a corregir problemes d'un sistema, ordinador o programa.
Monitoring: Procés o tècnica de controlar l'estat d'un conjunt de dispositius i aplicacions mitjançant eines de software.
Web App: Web App o aplicació web, es tracta una eina de software, localitzada en un servidor, que els usuaris finals fan servir mitjançant un navegador web.
Back-end: El back-end és una part d'una aplicació web o altre tipus de software que té interfície gràfica. El back-end és la part encarregada de la lògica de negoci del software.
Front-end: Part d'una aplicació web o altre tipus d'aplicació, que s'encarrega de la visualització de les dades de forma gràfica.
Api REST: Representational State Transfer, és una interficíe que permet que dos sistemes es comuniquin mitjançant el protocol HTTP, per tal d'intercanviar dades en formats XML o JSON.
Web Service: és una tecnologia que permet la comunicació, fent servir diferents protocols, entre màquines connectades a la xarxa.
Framework: Plataforma per al desenvolupament d'aplicacions software. Es tracta, d'una estructura conceptual que serveix com a suport i guia en el desenvolupament del software.
Interfície gràfica o d'usuari: software o part d'un software que realitza la tasca de mostrar l'usuari mitjançant gràfics, icones e imatges la informació del software i permet l'usuari interactuar amb aquest.
Git: Software de control de versions.
Agile Development: Metodologia de desenvolupament que defineix uns mètodes i pràctiques a seguir, basts en uns valors i principis.
SCRUM: Mode de treball que pertany a la metodologia Agile Development.
Sprint: Iteració de duració entre dues setmanes i 1 mes, format per diferents tasques.

APÈNDIX

A1. Diagrama de Components

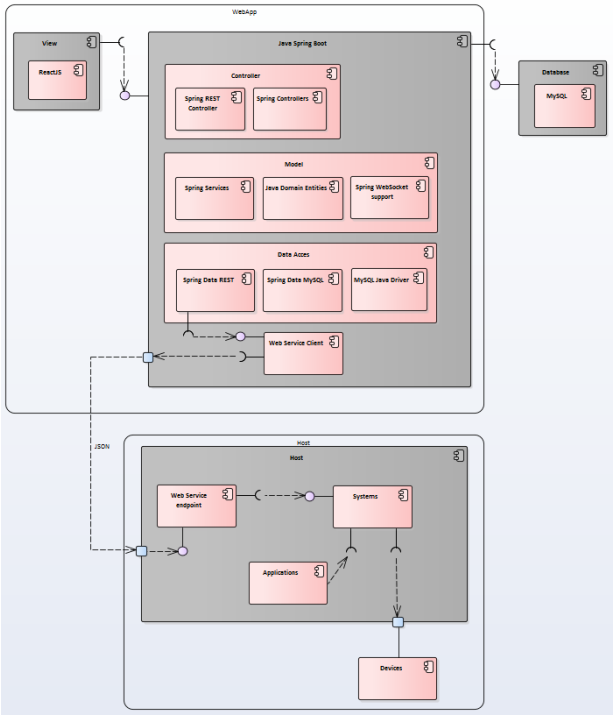
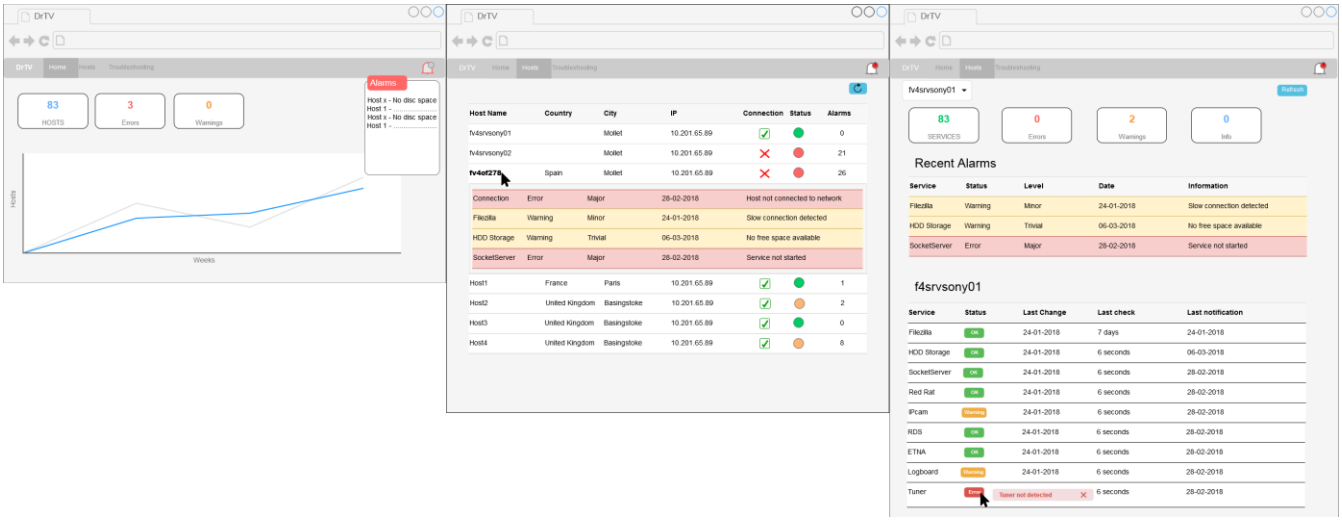


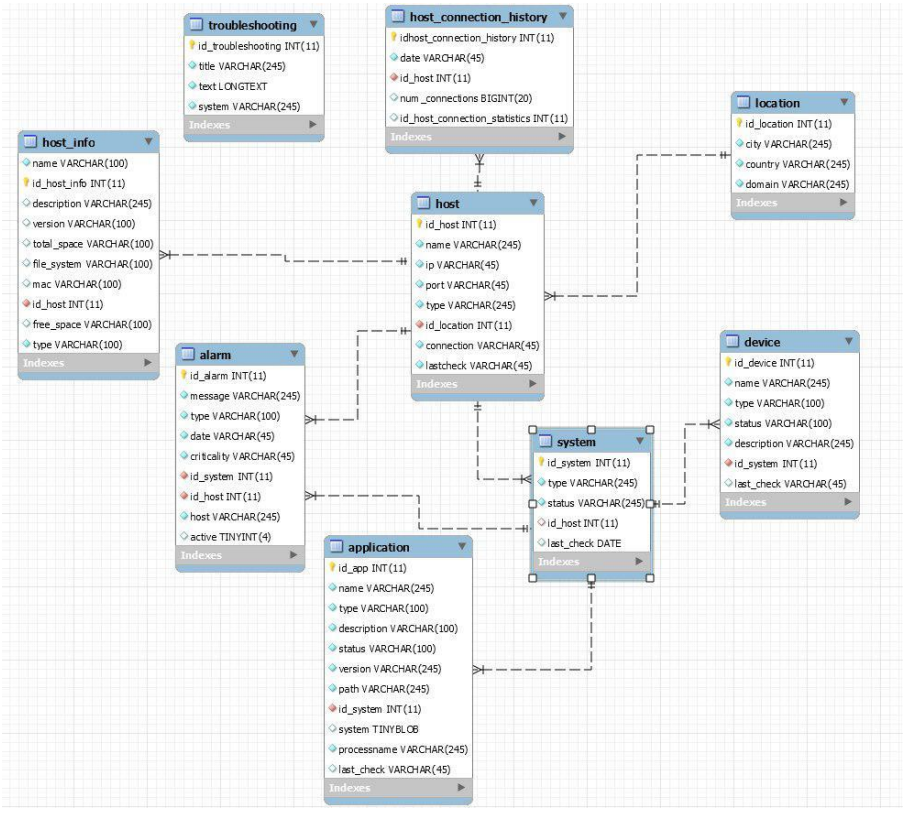
Diagrama de components representa com està dividit el sistema en diferents components i mostra les dependències dels components entre si.

A2. Mockups

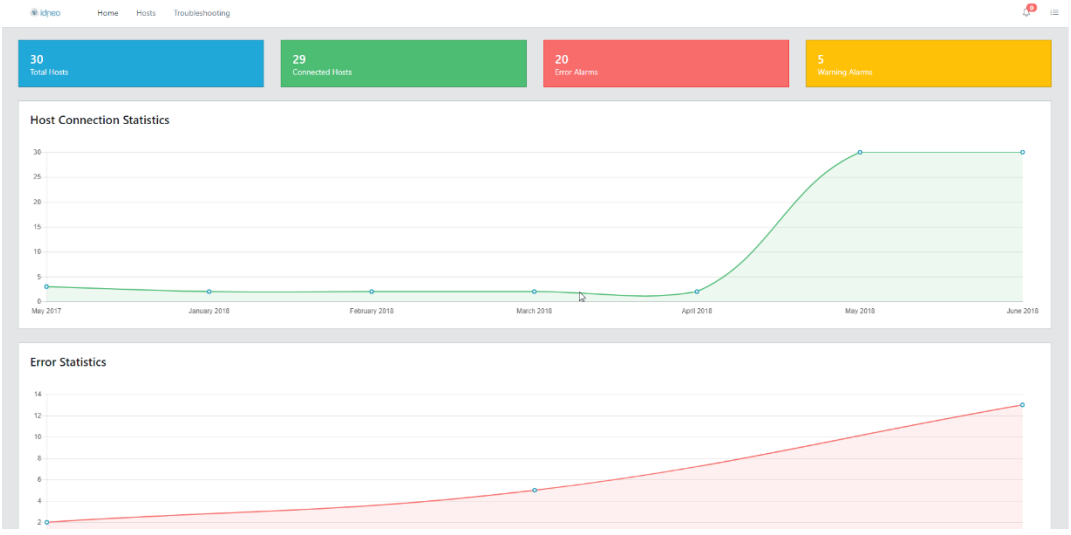


Mockups creats durant el procés d'anàlisi de requisits, es pot observar la pantalla inicial, el llistat de hosts connectats a la infraestructura IT i els detalls d'un host i el seu estat.

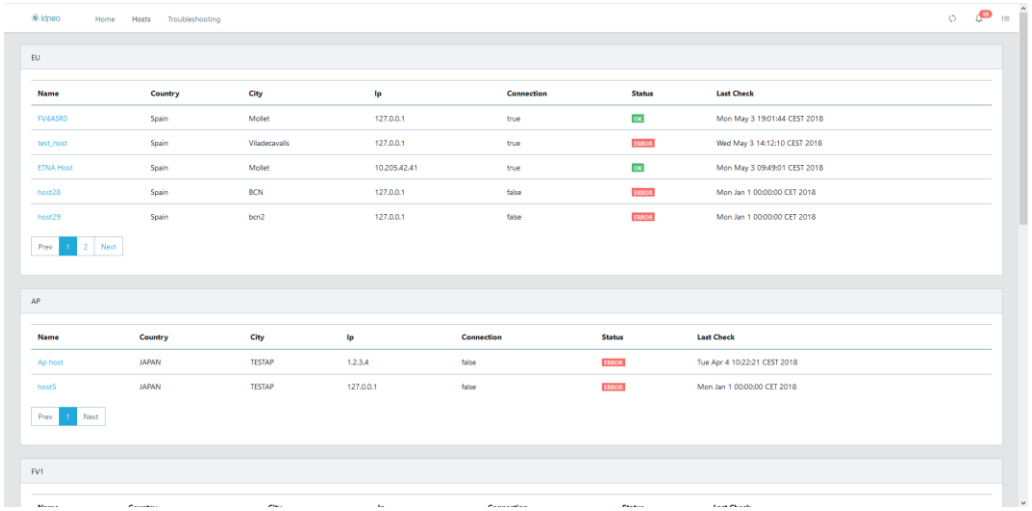
A3. DIAGRAMA DE CLASSES DE LA BASE DE DADES



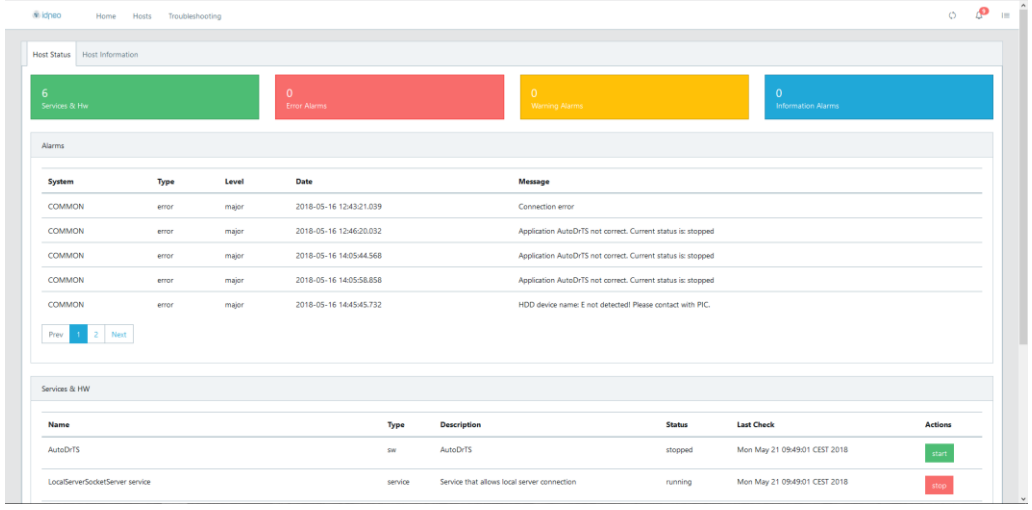
A4. FRONT-END



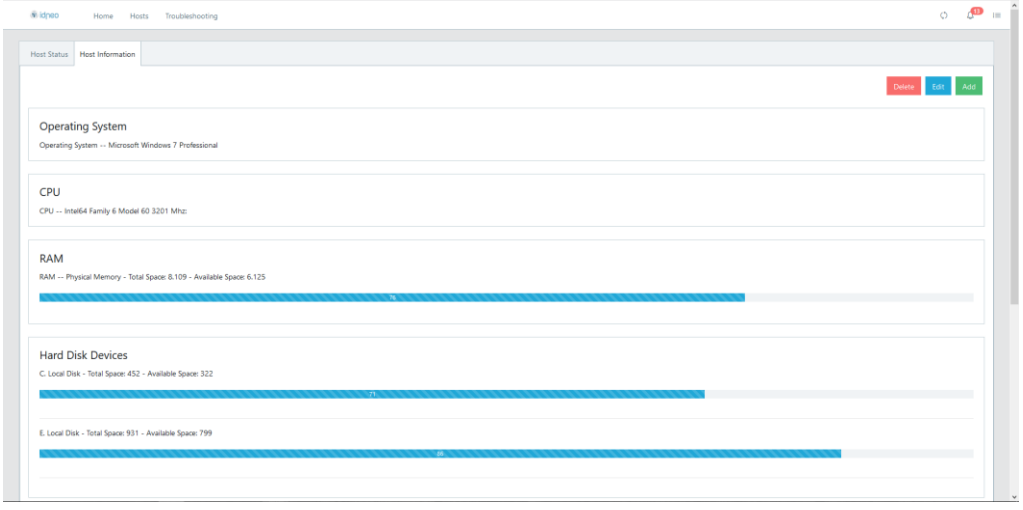
Pantalla home de l'aplicació on es pot veure un resum de l'estat del sistema i estadístiques de connexions i errors.



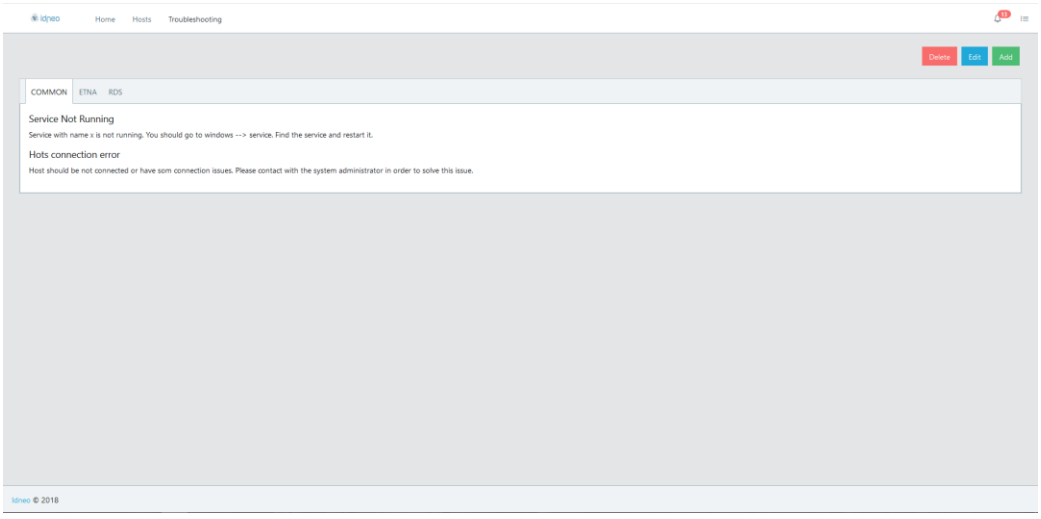
Pantalla "Hosts" on observem el llistat de tots els hosts classificats per domini, a més de característiques generals com localització, estat o última comprovació de l'estat.



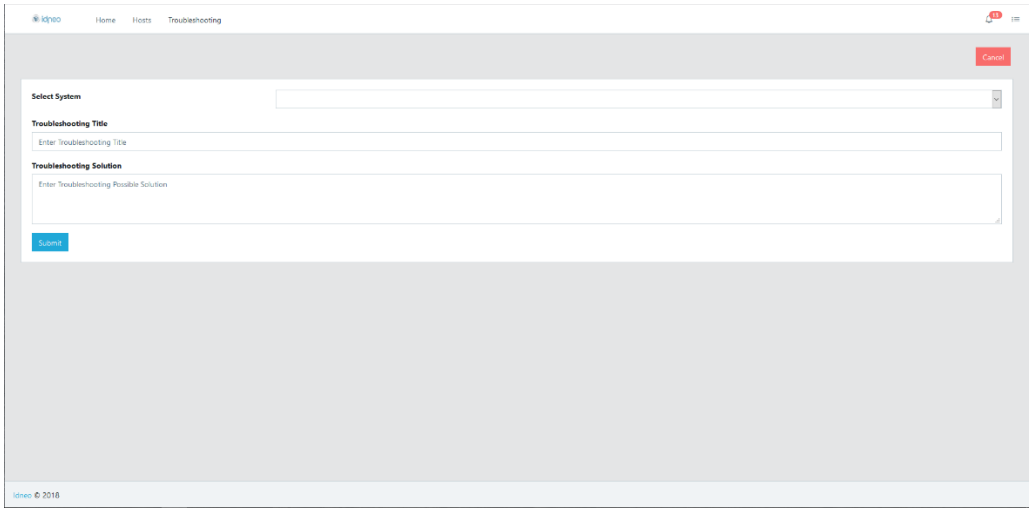
Pantalla d'un hosts específic, on podem veure en detall les darreres alarmes que han sorgit, l'estat d'aplicacions i dispositius, a més de poder accedir a una pestanya d'especificacions del host.



Informació detallada d'un host, a més de les especificacions es poden gestionar dispositius, aplicacions i serveis. Permet l'edició, eliminació i afegir-ne de nous elements.



Pantalla Troubleshooting on hi ha les possibles solucions que aportin els administradors. Tenen l'opció d'afegir noves solucions, editar existents o eliminar-les.



Formulari per afegir noves entrades al Toubleshooting.