CoKit: Control d'estoc de la nostra cuina.

Francesc Miralles Alsina

Resum—El projecte es basa en la creació d'una eina per controlar l'estoc alimentari que tenim a la nostra cuina i l'ajuda a realitzar compres futures a diferents supermercats. L'eina està formada per la part hardware, on controlem, mitjançant un lector de codi de barres amb una Raspberry Pi Zero, els productes que afegim i eliminem, i l'aplicació per a Android, on podem observar els productes que disposem en aquell moment, llistes de compres que podem realitzar amb el supermercat més econòmic per aquella compra, comparador de preus de diferents supermercats, i altres configuracions en l'apartat hardware. L'aplicació també és funcional sense disposar del lector de codis de barres.

Paraules clau— Control, estoc, raspberrypi, android, compres, preus, codis de barres, cokit

Abstract— The project is focused in creating a tool to control the food stock we have in our kitchen and help you to make future purchases in different supermarkets. The tool is made up of the hardware part, where we will control, by means of a bar code reader with a Raspberry Pi Zero, the products we add and eliminate, and the Android application, where we can observe the products we have in that one, shopping lists we can do with the most economical supermarket for that purchase, price comparison of different supermarkets, and other configurations in the hardware section. The application will also be functional without having the barcode reader.

Index Terms— Control, stock, raspberrypi, android, purchase, price, barcode, cokit

1 Introducció

Avui en dia la robòtica, l'automatització i la innovació és cada cop més freqüent. Poques coses de la vida quotidiana no estan unides a les noves tecnologies. Per això, a totes les cuines d'arreu del món trobem cada cop més, tecnologia entre els nostres aliments... neveres, microones, batedores, robots de cuina... [Fig.1] Però una de les coses importants abans d'utilitzar totes aquestes eines de cuina és la necessitat d'adquirir els nostres aliments.

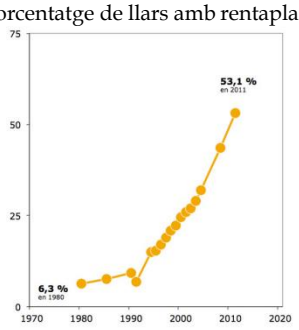


Figura 1: IDAE y Encuesta Continua de Presupuestos Familiares. Podem veure com la tecnología ha evolucionat a les feines de la llar.

Per això, aquest projecte consisteix a facilitar i incloure la tecnologia a la tasca de comprar aliments per la nostra cuina, ja sigui per l'ús personal com a serveis de restauració.

1.1 Objectius

Es vol ajudar a mantenir un estoc just i facilitar les compres dels productes, per això els nostres objectius són:

- Permetre controlar l'estoc dels productes que disposem a la nostra cuina. Mitjançant una aplicació per Android, podrem saber que disposem a qualsevol lloc amb connexió a internet.
- Poder veure el nostre historial de compres i ús dels productes, i així observar quins productes són els que utilitzem més o menys.
- Poder generar automàticament llistes de compres respectant el nostre historial. Així, mitjançant una heurística de temps o de diners, el programa ens llista un conjunt de productes que podrem fer una compra molt més optima.
- Calcular a quin supermercat ens sortirà les llistes generades de compra més econòmiques. Utilitzant els preus de les diferents webs dels supermercats, podrem obtenir els millors preus.
- Poder comparar preus individuals de productes concrets.

- *E-mail de contacte: cesc.miralles.alsina@gmail.com*
- *Menció realitzada: Computació*
- *Treball tutoritzat per: C. Alejandro Parraga (Computer Vision Centre)*
- *Curs 2017/18*

1.2 Estat de l'art

Durant els últims anys, l'increment de les compres online ha sigut molt alta, passant el 2010 a una mitjana de 831€ a l'any per persona a quasi 1200€ al 2016. [Fig.2]

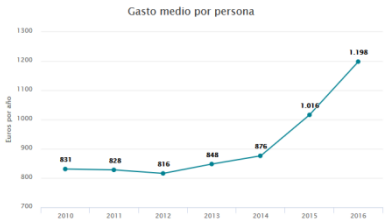


Figura 2: Ministerio de energía, turismo y agenda digital

Això ens permet deduir que la gent està cada cop més unida a la tecnologia i s'aprofita d'ella per a accions quotidianes. Antigament, els inicis d'aquest *boom* que vivim ara, hi havia un ambient de por o respecte a aquestes tecnologies al fet que pocs eren els que confiaven els seus diners amb fer les compres via internet. [1]

També, podem pensar que la gent cada cop té menys temps o no vol gastar aquest en anar a comprar, i que qualsevol ajuda a guanyar temps per a altres coses són ben vistes.

El fet d'anar a fer la compra s'ha tornat a una tasca que la gent cada dia rebutja més, i encara més si la compra a realitzar és alimentaria. Segons un estudi realitzat per la Universitat Complutense [2], del 2017 al 2018 ha augmentat un 10% les compres d'aliments mitjançant Internet. Per això, la necessitat d'una eina que ens permeti tenir la llista de compra feta, de no necessitar comparar preus de diferents supermercats i que el fet d'entrar al supermercat i fer la compra sigui la més ràpida possible és una necessitat que cada cop és més necessària.

Actualment podem trobar aplicacions per gestionar els nostres aliments, però totes les que hem trobat per a la gestió de l'estoc es posen manualment, però no tenim coneixement d'aplicacions que gestionin de manera automàtica llistes de compres, ni que es puguin afegir productes fàcilment mitjançant el codi de barres. També podem trobar alguna web o aplicació de comparació de preus de supermercats. En aquests s'ha de buscar un per un el producte i et mostra una petita descripció i els diferents preus que podem trobar en els supermercats.
En canvi, no hem pogut trobar cap aplicació que et generi llistes de compra automàticament.

L'aplicació per a Android "Llista de compres" és la més semblant i la més descarregada en aquest àmbit [3]. Aquesta eina ens permet crear manualment llistes de productes i poder comparar el preu total en diferents supermercats.
Segons la descripció de l'aplicació podem:

- Crear tantes llistes de compres com desitgem.
- Crear, modificar o eliminar els supermercats i els seus articles un cop creats.
- Copiar, parcialment o totalment, llistes d'un altre supermercat minimitzant així al teclejar nous articles.
- Introduir quantitats si ho desitgem.
- Configura mida i color del text i fons.
- Utilitzar tres tipus de llistes.
- Mantenir la pantalla encesa i la brillantor atenuat durant la compra.

En vista que no trobem una aplicació que ens permeti satisfer de les necessitats dels nostres objectius, hem creat una aplicació, Cokit, que ens permet facilitar la gestió dels nostres aliments i generar llistes automàticament, alhora comparar els preus a diferents supermercats.

A diferència de CoKit, Llista de compres ens permet crear llistes manualment i així guardar-les per compres futures, en canvi no genera les llistes automàticament, ni permet saber els productes que disposem. Cokit ens permet interactuar amb els codis de barres, en canvi *Llistes de compres* no disposa, sigui per software o per hardware, detectar productes mitjançant els codis de barres.

1.3 Planificació

El treball s'ha planificat en dies tot indicant les tasques que s'havien de dur a terme en cada període. Inicialment la planificació era lleugerament diferent però a causa dels canvis que van apareixent en el transcurs del projecte ha sigut canviada per tal que doni una visió més encertada de com ha sigut realment.

Tot seguit trobem la taula de la planificació:

Tasques			
Fase Inicial	14 dies	15/02/18	19/03/18
Reunió inicial	1 dia	15/02/18	15/02/18
Definició dels objectius	1 dia	16/02/18	16/02/18
Definició del projecte	2 dies	20/02/18	21/02/18
Preparació i instal·lació del hw i sw necessari	2 dies	23/02/18	24/02/18
Informe inicial	1 dia	10/03/18	10/03/18
Aprenentatge i enllaç del lector amb la RpiZ	3 dies	10/03/18	13/03/18
Test	1 dia	14/03/18	14/03/18
Desenvolupament de la App Android. Estructura del sw.	3 dies	16/03/18	19/03/18
Fase de Desenvolupament	83 dies	20/03/18	10/06/18
Extracció de preus a diferents webs de supermercats. Android/java.	3 dies	20/03/18	23/03/18
Comparador de preus.	1 dia	24/03/18	24/03/18

Android.			
Segona reunió	1 dia	11/04/18	11/04/18
Informe de progrés I	3 dies	18/04/18	21/04/18
Tercera reunió	1 dia	18/05/18	18/05/18
Informe de progrés II	4 dies	19/05/18	23/05/18
Desenvolupament de generació llista d'estoc i historial.	5 dies	24/05/18	29/05/18
Informe de progrés III	2 dies	25/05/18	27/05/18
Desenvolupament llistes generades.	8 dies	2/06/18	10/06/18
Configuracions Supermercats	1 dia	13/06/18	13/06/18
Actualitzar mètode extracció preus	2 dies	13/06/18	15/06/18
Desenvolupament Python (Raspberry Pi Zero W)	6 dies	14/06/18	20/06/18
Fase de Test & Resultats	11 dies	18/06/18	29/06/18
Test lector de codis	2 dies	18/06/18	20/16/18
Test aplicació Android	2 dies	20/06/18	22/06/18
Questionari	11 dies	18/06/18	29/06/18

2 METODOLOGIA

Per dur a terme tot el projecte s'ha seguit una estructura en 5 passos, aquests són els següents: *Anàlisi de requeriments, disseny, implementació, test i evolució.*

També s'ha intentat realitzar el treball utilitzant una metodologia àgil, concretament SCRUM, gràcies a l'adaptabilitat que dóna a l'hora de modificar objectius, a més de ser una metodologia molt comuna en el sector i es desitja simular un entorn professional de testing al màxim possible.

2.1 Part Hardware

En l'apartat hardware disposem de dues parts bàsiques:

1. Pistola lectora de codi de barres Welquic [4].Ens permetrà detectar els productes mitjançant el codi de barres rapidament. El lector envia el número llegit via usb. Altres alternatives podrien ser utilitzant la càmera del smartphone per detectar els codis de barres, però amb la pistola guanyem rapidesa, ja que quasi és instantani, i amb la càmera dependríem molt de la qualitat d'aquesta i la il·luminació.

2. Raspberry Pi Zero W[5]. L'encarregat d'executar el programa Python que ens permetrà llegir els codis proporcionats per la pistola de codi de barres i transferir aquests codis a la nostra base de dades.
   La versió de la Raspberry Pi ZeroW ens permet un estalvi en comparació a les altres versions, ja que el seu preu és més d'un 60% inferior a les versions estàndards. Aquest fet ens obliga a respectar els recursos d'aquesta placa, ja que també és bastant inferior a la Raspberry Pi 2/3.

Figura 3: Lector codi de barres i Raspberry Pi ZeroW

2.2 Part Software

Els objectius de l'aplicació per Android es divideixen en tres parts:

1. Controlar l'estoc que disposem actualment a qualsevol lloc on estiguem, gràcies a l'aplicació per sistemes Android.

2. Comparar els preus dels diferents supermercats que disposem d'un producte concret. Els preus son actualitzats per les diferents pàgines web dels supermercats.

3. Generar llistes de compres per data o preu total. L'aplicació, mitjançant l'historial d'entrades i sortides de productes, pot generar llistes de compra més eficients.

2.3 Eines Utilitzades

Per la realització de totes les tasques, hem escollit eines que tinguem a l'abast rapidament, per això quasi tot el software utilitzat és d'ús lliure i gratuït:

- Utilitza el sistema operatiu GNU/Linux **Raspbian**, d'ús lliure i dissenyat pel bon funcionament en Raspberry Pi. La nostra versió no parteix d'interfície gràfica per l'estalvi de recursos. [6]

- El llenguatge de programació de l'apartat hardware, el que ens permetrà llegir els codis, extreure la informació del producte i afegir/eliminar productes a la nostra base de dades, serà **Python 3**. Aquest llenguatge interpretat ens permetrà executar totes aquestes tasques esmentades en el menor línies de codi i així estalviar en recursos. En altres llenguatges que teníem com a alternatives, no disposen de llibreries per connectar amb la base de dades que utilitzem, Firebase, i el desenvolupament seria molt costós, tant per dificultat com a recursos. [7]

- La base de dades que ens proporcionarà la informació dels productes utilitzada serà **UPC Database** [Fig.4]. Una base de dades lliure i internacional[8]. Aquest servei ens proporciona eines per integrar la seva base de dades a diferents llenguatges de programació mitjançant una api. Hi ha alternatives a altres bases de dades, però la UPC ens permet integrar la base de dades a di-

versos llenguatges de programació i també ens permet afegir productes que no disposa.

Figura 4: Logotip de UPC Database

- L'IDE utilitzat per la part Python serà l'eina **Notepad++** [9], ja que ens proporciona moltes facilitats i és d'ús lliure. Hi ha diverses alternatives, però aquesta disposa de moltes facilitats en el desenvolupament i tenim experiència, ja que és el que més hem utilitzat.

- Pel desenvolupament de l'aplicació Android utilitzem l'eina de Google per aquesta finalitat, **Android Studio** [10]. Amb aquest IDE ens permetrà, mitjançant una relació entre XML i Java, crear l'app que necessitem. És gratuïta amb llicència Apache 2.0. Hi ha altres alternatives, la més utilitzada, després d'Android Studio, és Eclipse amb els afegits per Android. Per tema d'estabilitat, facilitats, compatibilitats i per experiència, hem utilitzat la proporcionada per Google.

- Per a la nostra base de dades, on guardarem l'historial i els preus dels productes obtinguts de les webs dels supermercats, serà amb la plataforma **Firebase** [11]. Aquesta potent eina ens proporciona una base de dades no relacional, en temps real i back-end. Fins a una certa capacitat i un limit de tràfic, el portal és gratuït. Una dels avantatges respecte a altres tipus de base de dades són:

    - Gran compatibilitat amb Android Studio.

    - Base de dades en temps real, ens permet veure els resultats automàticament des de l'aplicació sense refrescar o tornar a carregar l'aplicatiu.

    - Base de dades noSQL. Ens permet tenir una base de dades molt més escalable i flexible que la SQL. Al no treballar en taules, modificar, afegir claus o valors és molt més trivial.

    - Gratuït fins a un cert límit de tràfic. Després es paga segons la dimensió del nostre projecte.

    - Back-end on podem gestionar i monitoritzar part de la nostra aplicació. Usuaris actius, errors o problemes dels usuaris, gestió de comptes d'usuari, compatibilitat en inici de sessió des de altres plataformes (Facebook, Google+, twitter...).

    - Fiabilitat i garantia de Google. Firebase va ser adquirida per l'empresa de Larry Page a l'any 2014. [12]

- **Microsoft Office** ha sigut l'eina utilitzada per tot l'apartat de documentació, ja que és molt potent per la realització d'aquesta tasca i tenim disponible tots els paquets gràcies a les llicències gratuïtes que ens proporciona la universitat. Altres alternatives que ens permetrien resultats semblants podria ser OpenOffice o LibreOffice. Al disposar de les llicències de l'eina de Microsoft i per l'experiència d'aquest programa, vam escollir aquest últim.

3 DESENVOLUPAMENT

3.1 Part Raspberry Pi ZeroW

En aquesta part de desenvolupament consistirà a la realització i la implementació en Python3 per enllaçar la pistola lectora de codi de barres i la placa Raspberry Pi ZeroW.

En un inici, estava planificat que el llenguatge a utilitzar fos Python 2.7, ja que la majoria de treballs realitzats a la universitat era amb aquesta versió de llenguatge. Però el moment d'implementar l'Api de UPCDatabase, aquesta només està disponible per a Python3, així que, per no complicar la implementació, i el fet que la diferència entre les dues versions són escasses, es va decidir que la implementació d'aquest apartat fos amb Python3. [Fig.5]

Figura 5: Exemple d'obtenció d'informació de la BarcodeDatabase

Un cop obtenim la informació del producte, el codi envia aquesta informació a la nostra base de dades Firebase, on guarda el codi de barres, per poder obtenir aquell producte en un futur sense haver de tornar a fer una consulta a la BarcodeDB, el nom del producte, escurçant la cadena de caràcters amb les primeres paraules més rellevants, incrementar o decrementar el número d'estoc segons si es vol afegir o extreure, i per finalitzar, la data de la transacció.

3.2 Part aplicació Android

Per desenvolupar una aplicació Android, hem de tenir clar que formen part de tipus de llenguatge:

- XML, un meta-llenguatge definit per marques. Aquest fitxers xml ens permetrà definir la part gràfica del software, l'especte que tindrà la nostra aplicació.

- Java, llenguatge orientat a objectes que donarà forma tot el funcionament de la nostra aplicació [13]. El llenguatge està format per classes i herències.

Primer de tot vam definir una interfície per la nostra aplicació, utilitzant un menú desplegable mitjançant un botó a la part superior esquerra [Fig.6], molt característic per les aplicacions d'Android. També incorporem un botó flotant a la part inferior dreta que ens permetrà anar amb un fàcil accés al comparador de preus individual (sols un producte contret).

Un cop ja teníem el disseny estructurat i funcional, vam implementar la part de comparador de preus individual, ja que aquesta mateixa funció serà reutilitzada per la generació de preus de les llistes.

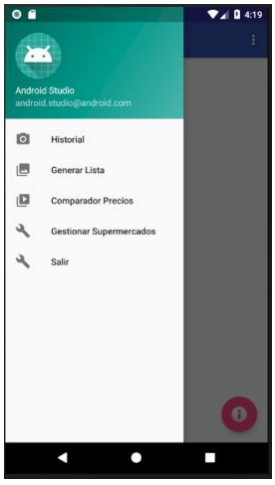


Figura 6: Disseny utilitzat per l'aplicació. A l'esquerra es desplega un menú vertical que ens permét interactuar amb l'aplicació

Per realitzar aquesta tasca, hem implementat amb el llenguatge Java, una consulta a les diferents webs de supermercats on aconseguim tot el codi html de la pàgina del o dels productes que busquem i així buscar mitjançant les etiquetes html el preu o preus que necessitem. En aquest apartat, hem incorporat una caixa de text per buscar el producte mitjançant el nom, ja sigui el genèric com pot ser "leche" o per marca.

Un cop realitzem la cerca, ens va apareixent els preus dels diferents supermercats. Podem observar que depèn de la web del supermercat pot tardar més o menys, però en conjunt el temps d'espera és més elevat del que ens podíem imaginar (al voltant d'uns 3 segons per supermercat on en conjunt sumen uns 15 segons). Per això vam pensar a integrar aquests preus a la nostra base de dades i així poder obtenir-los molt més rapidament. Per tenir aquests preus actualitzats periòdicament, guardem els preus amb la data del dia actual, i si el cop que volem treure el preu d'aquell producte ha passat més d'un cert temps, per exemple una setmana, torna a buscar el preu a la web i l'actualitza amb la data també actualitzada. D'aquesta manera la majoria de cops traurem el preu amb un temps molt més inferior (un 95% més ràpid) i assegurar-nos que aquell preu és actualitzat. [Fig.7]

Figura 7: Comparador de preus individual. Disposem de 4 supermercats.

Després de tenir uns quants problemes en la integració de la base de dades Firebase en el codi Python, ja que la instrucció "push" ens afegía uns nodes intermedis amb uns tokens aleatoris, un cop canviant aquestes consultes i la part Android pogués detectar els productes en estoc, ens posàvem a desenvolupar la pantalla principal de la nostra aplicació on es mostra la llista de productes. [Fig.8]

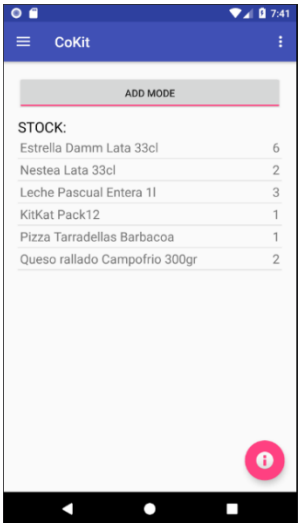


Figura 8: Llista de l'estoc (pantalla principal).

També vam afegir un botó a sobre la llista, que configura el mode de la pistola de codi de barres. Si tenim activat el mode "Add Mode", la pistola afegirà el producte a la llista un cop detecti un codi de barres, si en canvi, tenim el mode "Remove Mode" activat, la pistola restarà el producte a la llista. Al costat d'aquest, disposem d'un camp de text per afegir el número de productes que volem afegir o eliminar. Facilita a l'usuari quan hagi d'afegir packs de productes o hagi d'utilitzar una gran quantitat d'un producte que disposi.

Per poder generar les llistes automàticament, necessitem guardar i gestionar l'historial d'entrades i sortides. Per això la nostra base de dades disposarà de totes aquestes transaccions. L'usuari té a la seva disposició una pantalla que li mostra tots aquests moviments, juntament amb la data i el número que ha afegit o restat a l'estoc del producte. En aquest apartat fem un recorregut a la base de dades i anem afegint cada moviment en un item a la llista juntament amb els valors d'aquell moviment.

Ara que ja tenim en funcionament el llistat i la configuració de la pistola, només ens queda desenvolupar l'algoritme per generar les llistes automàticament. Per això haurem de calcular una heurística que ens permeti poder generar les llistes més idònies.

Ens interessa saber quants dies transcorren per ser consumit cada producte segons el seu historial. Per això, el valor significatiu serà els moviments que resten l'estoc. De la base de dades obtindrem la data, el producte, el número de moviment i el tipus, si és per afegir o eliminar. Per aquest cas farem un recorregut per tots els registres i guardarem només els que són del tipus eliminar. D'aquests farem una mitjana ponderada dels dies que tarda un producte a ser consumit. [Fig.9]

$$\bar{X} = \frac{(DiesTranscorreguts * NumMoviment) + EstocAnterior}{EstocTotal}$$

Figura 9: Càlcul per obtenir la mitjana ponderada dels dies per producte consumit

Un cop obtenim aquest valor per a tots els productes, els botons per generar les llistes s'activaran, així ens assegurem que l'aplicació no generi llistes sense disposar de tots aquests valors.A l'interfície disposem de dos botons, un per generar les llistes posant un límit de diners, i l'altre limitant els dies que volem que ens duri la compra. Just al costat dels botons tenim dos camps de text per afegir els valors de restricció.

Per a la generació de llistes segons els dies, fem un recorregut per obtenir el producte que consumim més diàriament i anem afegint a la llista, un cop afegim un producte comprovem que no sobrepassem el valor de dies, si no és així repetim el procés. Pels productes ja afegits, a la comprovació per obtenir el producte que consumim més ràpidament, li afegim els temps de l'anteriors, així no obtindríem llistes només amb aquests productes. [Fig.10]

A la versió de les llistes per diners és molt semblant a l'anterior però fent la comprovació amb el valor del càlcul anterior juntament amb els preus obtinguts. Crearem 4 llistes diferents, un per cada supermercat. Si en un producte no troba el preu a la base de dades, aquest haurà d'obtenir-lo de la pàgina web de l'establiment, i això generarà un temps d'espera superior, però per a futures operacions ja disposarem d'aquest valor sense necessitat d'esperar. Quan tinguem les quatre llistes generades on el total de diners hauria arribat a l'establert per l'usuari, mostrarem la que més productes obtindríem juntament amb el supermercat escollit.

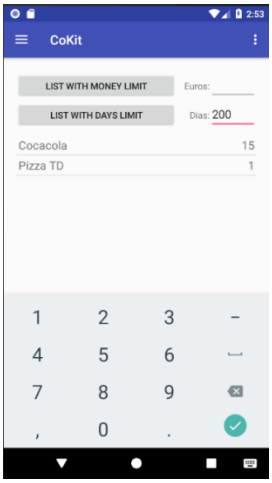


Figura 10: Interfície de la pantalla de generació de llistes. Exemple d'una llista generada amb 2 productes posant el límit en 200 dies.

Si alguna de les diferents webs dels supermercats no troba el producte del que necessitem saber el preu, estableix el preu més alt dels altres supermercats. D'aquesta manera obtindríem un preu final de compra més aproximat al que trobarem realment i quasi sempre els resultats no seràn més baixos que el real.

Per a usuaris que no disposin d'algun supermercat a curta distància, s'ha creat un menú de configuració per establir quins supermercats volem que mostri l'aplicació. [Fig.11]

Figura 11: Interficie que permetrà a l'úsuari configurar els supermercats utilitzats

Aquesta configuració serà l'única que no es guardarà a la base de dades, ja que només la necessitarà l'aplicació. D'aquesta manera no ocuparà espai a la base de dades i l'ocuparà al smartphone utilitzat, que aquest valor és insignificant. Per això utilitzem la interfície proporcionada per Android, SharedPreference [14], que serveix pel propòsit que volem, guardar configuracions de la nostra aplicació. Aquesta informació no és volàtil, i sempre obtindríem la configuració sense necessitar conexions a xarxes externes.

3.3 Base de dades

La base de dades d'aquest projecte és una peça fonamental, ja que ens permetrà la comunicació entre la pistola de codi de barres connectada a la Raspberry Pi Zero amb l'aplicació d'Android. Per això escollit la millor tipus de base de dades va ser una tasca molt complexa.

Com s'ha comentat anteriorment, la bdd escollida va ser Firebase de Google. Al no ser una base de dades SQL, el format que guarda els registres són significativament diferents de la de taules. El format és molt semblant als arbres, on nodes pares penja'n fills. [Fig.12]

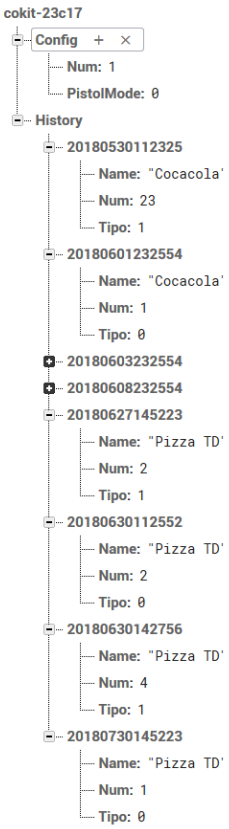


Figura 12: Estructura de la base de dades amb productes i configuracions d'exemple.

La nostra base de dades està estructurada en dos parts fonamentals:

- "Config": Aquí disposarem de la configuració de la pistola de codis de barres que gestionarem des de l'aplicació. Com hem comentat anteriorment, la base de dades és el pont per pasar dades i configuracions entre la RPi i l'aplicació Android. En el nostre cas tenim 3 principals nodes, si la pistola afegeix o elimina, quants elements gestiona per cada moviment i els supermercats que té l'usuari com a actius.

- "History": Aquí la Rpi, mitjançant Python3, guardarà tots els moviments que faci amb la pistola. Guardarà la data actual en el node pare i el producte, el número i el tipus de moviment en els nodes fills.

4 TESTEIG DE RESULTATS

4.1 Tests

Per obtenir resultats en tan poc temps sense disposar d'usuaris de tests (només disposem d'una pistola lectora de codis de barres), hem hagut de passar unes proves de caixa blanca i caixa negra [15] per obtenir resultats i problemes que pugui tenir. Per això hem posat a prova el software mitjançant aquestes pràctiques:

1.000 dades en el historial (afegit manualment des de la base de dades)

A l'afegir dades a l'historial no hem tingut cap problema, ni per integritat de les dades ni per temps d'accés a la base de dades. En generar les llistes sí que s'ha notat un temps major que amb menys dades. Exactament 12,8 segons més que amb 10 dades d'historial.

10.000 dades en el historial (afegit manualment des de la base de dades)

A l'afegir dades a l'historial no hem tingut cap problema, ni per integritat de les dades ni per temps d'accés a la base de dades. En generar les llistes sí que s'ha notat un temps major que amb menys dades. Exactament 180,7 segons més que amb 10 dades d'historial.

Diferència en obtenir preus de la base de dades Firebase i la mitjana de webs dels supermercats

Firebase: 0.7 segons

Web de supermercats: 9.1 segons

En aquesta prova podem obtenir l'importància de tenir els preus guardats a la nostra base de dades. La diferencia de temps és molt superior (1300%).

Afegir producte sense informació a la UPCDatabase

L'aplicació Python de la Raspberry Pi W es para en una excepció i no afegeix el producte. Un cop afegit a la base de dades ja funciona en normalitat.

Generar llistes amb productes que no s'han consumit mai

Com s'ha comentat anteriorment, el càlcul que fa per obtenir les llistes generades, utilitza els valors de consum en l'historial. Si provem de generar una llista amb algun producte sense consumir, l'aplicació no pot generar i es queda en bucle infinit. Aquesta part ja està arreglada.

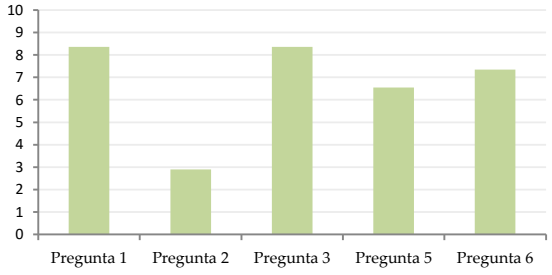
4.2 Qüesionaris de satisfacció

Per millorar i trobar problemes en el projecte, s'ha formulat un qüestionari on diferents usuaris de l'aplicació han

contestat després d'utilitzar-la durant uns dies. Les preguntes a contestar i els seus resultats són les següents:

1. T'ha semblat útil l'aplicació? Resultat: 8,35/10
2. Has tingut molts problemes/errors? Resultat: 2.9/10
3. L'hi veus futur? Resultat: 8,35/10
4. Havies utilitzat mai altres aplicacions semblants? Resultat: (SI: 4 NO:12)

5. Com qualificaries la interfície de l'aplicació? Resultat: 6,55/10
6. Com qualificaries la velocitat? Resultat: 7,35/10
7. Quant estaries disposat a pagar per l'aplicació i la pistola de codi de barres? Resultat: 28,55€



Usuaris que han respòs el qüestionari: 20

Podem observar que els resultats són satisfactoris, ja que les preguntes 1 i 3 és important que el resultat sigui alt, i en el nostre cas s'ha obtingut una mitjana de 8,35 sobre 10. En canvi, en la pregunta 3 era important obtenir un resultat baix, ja que mesurem les falles que han tingut amb l'aplicació. El resultat d'aquest últim ha sigut de 2.9 sobre 10. Els problemes que han tingut han sigut redactades i en moltes d'aquestes ja estan arreglades. En la quarta pregunta, on la resposta era binaria, veiem que no hi ha moltes aplicacions similars i/o no són utilitzades.

A la cinquena pregunta, on tracta la interfície, la nota no és molt alta, ja que el projecte no ha tractat aquest tema com a punt fort. Els enquestats han vist que l'aplicació és molt intuïtiva i fàcil d'utilitzar, però comparada amb l'estètica d'altres aplicacions, li falta color i bons dissenys. En la velocitat de l'aplicació l'han qualificada amb una puntuació molt alta el moment de generar llistes i a interactuar amb l'aplicació, però en el moment de recuperar els preus de les pàgines dels supermercats, aquest temps queda molt diferenciat amb els altres.

Com a última pregunta, tots els enquestats posarien un preu molt baix només a l'aplicació, ja que aquestes no acostumen a tenir preus molt elevats, En canvi, entenen que l'apartat de hardware influeix molt en el preu. La mitja dels preus proposats ens dóna un resultat de 28,55€, un preu ideal segons els enquestats seria de 29,95€.

5 CONCLUSIONS I LÍNIES FUTURES

Gràcies a la potència que actualment tenim en els smartphones hem pogut realitzar una aplicació, que juntament amb el lector de codi de barres, ens permet controlar fàcilment els productes que disposem. En el llarg del projecte han anat sorgin problemes on els hem anat solbentan en el procés de desenvolupament, com també han sorgint molt bones idees, ja siguin per afegir-les en l'aplicatiu com llistar-les per incloure-les en un futur.

La part que ens ha causat més problemes ha sigut la integració de l'aplicació Android i la part Python a la base de dades, ja que les consultes en la BDD no són iguals en els dos sistemes utilitzats.

En les reunions de seguiment i el fet de compartir la idea amb altres persones, han servit per trobar altres mercats que en un inici no estaven contemplats. Com pot ser al servei de restauració com poden ser restaurants, bars, hospitals... on les quantitats són bastant superiors i les compres donen molta més importància que en un habitatge. Per sort, l'aplicació no ha de canviar en res per ser utilitzades en els dos àmbits.

També han anat sorgint ideas per augmentar la facilitat i la comoditat per utilitzar l'aplicació. Aquestes idees són:

- Afegir tant el lector de codi de barres i el software de gestió a les neveres intl·ligents. Aquestes neveres ja disposen de pantalla tàctil i no seria molt complicat introduir CoKit en aquestes. La dificultat estaria en temes burocràtics amb les marques dels electrodomèstics i els supermercats.

- Canviar els codis de barres per la tecnologia RFID [16]. En alguns establiments, com pot ser Decathlon, ja incorpora aquest sistema a tots els seus productes i gràcies a aquesta tecnologia, els productes són detectats a distàncies molt més grans que els codis de barres. Tan si l'aplicació és usada en un smartphone o està anidada a les neveres, la tecnologia RFID ens donaria una comoditat i una rapidesa molt alta, ja que només passant el smartphone per sobre dels productes ja s'afegirien a la llista, en canvi en una nevera, podria detectar els productes que estan a dins.

AGRAÏMENTS

M'agradaria agrair tot l'esforç per la guia i el seguiment obtingut pel meu tutor del treball, C. Alejandro Parraga. En moltes parts, principalment de la documentació, no haguessin sigut possible sense l'ajuda obtinguda.

D'altra banda, també m'agradaria agrair a tots els professors i companys que durant aquests anys a la universitat m'han ajudat a formar-me en aquesta professió. Part d'aquest treball és gràcies a ells.

BIBLIOGRAFIA

[1] La historia detrás del boom de las ventas online. [Online] https://www.infobae.com/especiales/2017/11/02/la-historia-detras-del-boom-de-las-ventas-online/

[2] La mitad de los compradores por Internet ya adquiere alimentos vía "online". [Online] http://www.lavanguardia.com/economia/20180411/44245953 4398/compra-internet-comercio-electronico-alimentos.html

[3] Llista de compres. Android Play. [Online] https://play.google.com/store/apps/details?id=com.regilog.shoppinglist Visitat: 22/03/18

[4] Welquic USB Barcode Scanner Handheld Scanner. [Online] http://www.welquic.com/usb-barcode-scanner.html Visitat: 17/02/18

[5] Raspberry Pi Zero W. Raspberrypi.org. [Online] https://www.raspberrypi.org/products/raspberry-pi-zero-w/ Visitat: 17/02/18

[6] Raspbian. Descargar distribució. [Online]. https://www.raspberrypi.org Visitat: 23/02/18

[7] Python 3.0 Release. [Online] https://www.python.org/download/releases/3.0/ Visitat: 01/03/18

[8] UPCDatabase. Informació, busqueda i API. [Online] http://upcdatabase.org Visitat: 20/02/18

[9] Notepad++. [Online] https://notepad-plus-plus.org/ Visitat: 15/02/18

[10] Android Studio. Informació i descarga. [Online] https://developer.android.com/studio/index.html?hl=es-419 Visitat: 23/02/18

[11] Firebase. Informació, registre i manteniment. [Online] https://firebase.google.com Visitat: 23/02/18

[12] Firebase is Joining Google!. [Online] https://firebase.googleblog.com/2014/10/firebase-is-joining-google.html

[13] Reading a web page in Java. [Online] http://zetcode.com/articles/javareadwebpage/ Visitat: 20/03/18

[14] SharedPreferences. Android Developers. [Online] https://developer.android.com/reference/android/content/SharedPreferences Visitat: 13/06/2018

[15] Black-box vs White-box Testing. [Online] https://technologyconversations.com/2013/12/11/black-box-vs-white-box-testing/ Visitat: 19/06/2018

[16] Tecnologia RFID. [Online] http://www.dipolerfid.es/es/tecnologia-RFID Visitat 26/06/2018