TPVCloud
terminal de punto de venta *online*

Memoria del Proyecto Final de Máster

Máster Aplicaciones Multimedia

Itinerario profesional

Autor: Carlos Javier Garde Marí

Consultor: Sergio Schvarstein Liuboschetz

Profesor: David García Solórzano

Fecha de entrega: 20 de Enero de 2015

Créditos/Copyright

Al maravilloso e inolvidable Spectrum 16 KB.

Agradecimientos

Maldito “Subi” que me enseñaste a aprender a buscar más allá de los típicos tópicos.

Juan, Vero y Dani, siempre estáis.

Hi mom! You’re always on my mind, thank you for teach me that education brings you wisdom.

Abstract

This project is a Point of Sale software in the cloud, that is to say that a website have been developed with all the features of a typical local installation. So, basically, a seller can access to this software from everywhere and from any device (such a tablet).

Resumen

Este proyecto es un Terminal Punto de Venta en la nube, lo que quiere decir que es una *web* desarrollada con todas las características de una típica instalación en local. Así que, basicamente, un vendedor puede acceder a este programa desde cualquier lugar y desde cualquier dispositivo (como una tableta).

Keywords

pos, point of sale, sales management, SME accounting, SME sellings, recording sells, html5, cloud software, web application

Paraules clau

tpv, erminal punto de venta, gestión ventas, contabilidad PYME, ventas PYME, registro ventas, html5, programa en la nube, aplicación *web*

Notaciones y convenciones

Será usada la siguiente notación:

- Cursiva: para palabras en inglés, tales como *cloud*
- Cita bibliográfica, indicada con un número entre corchetes en estilo superíndice: concepto[10]
- Comillas: para nombres propios de algunos servicios, como “Openshift”

Índice

Figuras y tablas

Índice de figuras

Índice de tablas

Capítulo 1: Introducción

1. Introducción/Prefacio

La principal justificación de este proyecto es que el autor ha recibido varias veces peticiones de conocidos que poseen pequeños negocios, de poder disponer de una herramienta Terminal Punto de Venta[1] (de ahora en adelante TPV) más adaptada a ellos (no herramientas muy generalizadas que al final son muy complicadas de usar) y a unos costes que no sean tan elevados como los que hay en el mercado.

Normalmente, la compra de un terminal adaptado (no es más que un ordenador con pantalla táctil y un cajón para el dinero) suele implicar un gasto aproximado de unos mil euros. A esta cantidad se debe sumar el gasto del *software* TPV (que puede llegar a unos seiscientos euros anuales).

Sumemos a esto, los problemas aleatorios como que el *hardware* falle y se puedan perder los datos. Además de que suelen ser programas instalables, con limitaciones de sistema operativo y que muchas veces pueden requerir la asistencia de un técnico informático, teniendo que desembolsar aún más dinero.

Así que uno se plantea: ¿y si en la actual explosión de *SaaS*[2] (*software as a service*) diéramos paso a democratizar el acceso a estas herramientas con modelos *freemium*[3] (herramientas gratuitas en el uso básico y pagar por tener acceso a más características)? Es un nicho casi sin explotar, apenas hay competencia en las TPV e Internet es parte del día a día de la sociedad.

Cierto es que un potencial problema de desconfianza será el asunto de la privacidad de los datos en la nube, con el cual se tendrá que lidiar.

Además, este proyecto se justifica porque engloba varias facetas que el autor desea explorar y que las considera necesarias para su futuro profesional después de las diferentes asignaturas cursadas en el máster y después de dos años de haber tenido una empresa propia de desarrollo *web*. Luego de investigar en varias tecnologías, lenguajes y metodologías, desea poder aplicar lo descubierto en un proyecto de una forma más libre, sin las presiones del mercado laboral.

Tal como se contempla el proyecto, se visualiza como un triángulo de tecnología, gestión y posicionamiento.

En cuanto al posicionamiento, es al vértice que se le dedicará menos tiempo. Pero no por eso deja de ser interesante ver cómo aplicar técnicas para ver el funcionamiento del posicionamiento natural a lo largo de varios meses (más allá de la duración de esta asignatura) sin, de nuevo, la presión de clientes que desean que su página salga ya mismo en la primera posición sin pagar un posicionamiento *SEM*[4].

Por lo que se refiere a la gestión, es una necesidad saber cómo gestionar de una manera eficiente un proyecto desde su nacimiento hasta su entrega. La metodología ágil *Kanban*[5] se incluye en la

gestión de este trabajo y se considera que es la idónea para una trabajo individual. Ésta ha sido probada en los últimos meses y, realmente es una herramienta sencilla y con un resultado productivo sorprendente.

Por último, la tecnología. Sin lugar a dudas, poder hacer un proyecto con tecnologías punteras, o que están de moda, (tanto a nivel visual como de código) que se considera que pueden ayudar a dar un salto profesional importante, es un punto muy atractivo. La vida del programador ya no es "*.NET"*, "*Java"*, "*Cobol"* o "*PHP"*. Gracias a los servicios *Cloud* (que permiten usar casi cualquier lenguaje), la generalización y popularización de lenguajes como "*Ruby"*, "*Python"*, "*javaScript"*, "*Scala"*, etc están revolucionando el desarrollo *web* y se están haciendo con un hueco importante en el mercado profesional.

El lenguaje principal y que dará forma a este trabajo será "javaScript", ya que en este proyecto solamente se desarrollará en el *front-end*[6] debido a lo estudiado en el máster: desarrollo *web* y usabilidad.

Se dará un especial valor a la usabilidad para que una página sea usada sin tener que dedicar muchos segundos a pensar dónde encontrar las cosas.

En definitiva, es un proyecto sumamente interesante y atractivo para poder mejorar y profundizar en un mejor conocimiento del desarrollo informático.

2. Descripción

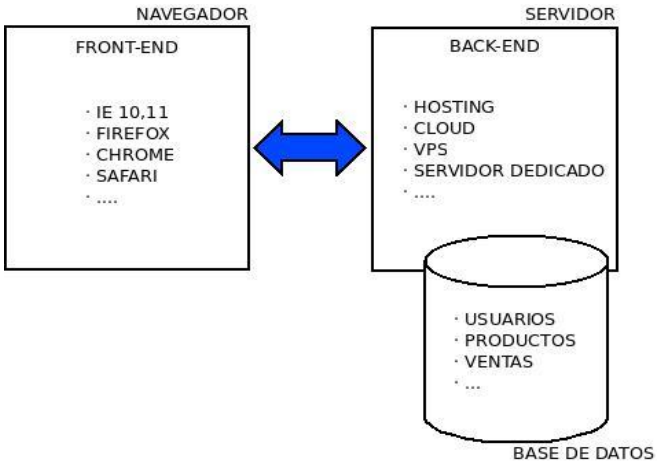
Se va a implementar un servicio (una página web *responsive*) que pueda usar cualquier persona desde varios dispositivos (ordenadores con diferentes resoluciones o tabletas) que posea un negocio de venta de productos/servicios al público en general y que desee registrar las ventas y clientes. Es un típico "Terminal Punto de Venta" que se puede hallar en la mayoría de pequeñas y medianas empresas, pero *online*, sin instalaciones, sin problemas de poder acceder dependiendo del estado del terminal físico y teniendo la seguridad que los datos no serán perdidos.

En la actualidad un *software* "TPV" necesita una instalación en máquinas locales, con unos gastos elevados (entre ordenadores que se venden con supuestas características necesarias para que función correctamente el "TPV", instalación técnica y pagos anuales por las licencias del *software*). Además, se corre el peligro de poder perder los datos si no se dispone de una cultura de ir salvándolos (y la mayoría de personas que poseen un pequeño negocio no la tienen).

Un sistema actual de TPV tiene más o menos la siguiente interfaz (en la cual se pueden hallar las principales características):

El proyecto no plantea ningún medio de pago ni pasarela ni conexión con un algún elemento externo que contenga dinero (típico cajón de dinero en efectivo o "TPV" físico para tarjetas que proveen las entidades bancarias).

En lo que se refiere a la página *web* a realizar, la arquitectura sería la siguiente:



En el dibujo anterior se puede observar como sería el flujo de trabajo desde el navegador a un servidor y una base de datos. Como se ha comentado anteriormente, este trabajo solamente se centrará en el desarrollo *web* en el navegador y no se implementa nada en el lado servidor.

Se optará por usar "HTML5", "CSS3" y "javaScript". No se plantea el uso de cualquier otra tecnología, pues es necesario que sea compatible con la mayoría de navegadores modernos y los dispositivos móviles existentes.

Cabe decir que no se va a poder asegurar la compatibilidad con viejos navegadores (o navegadores que nunca han sido muy compatibles con "HTML5"). En este último caso se estudiará la posibilidad de usar una biblioteca "javaScript" para la retro compatibilidad de elementos nuevos de "HTML5" con viejos navegadores. También, se intentará usar las características de "HTML5· más novedosas como implementación de etiquetas semánticas, almacenar datos en el navegador, etc... Es decir, poder hacer un uso profundo de las novedades que proporciona.

Siguiendo con la parte visual, también se plantea el uso de herramientas para la visualización como Bootstrap 3[7]. Estas herramientas facilitan mucho el desarrollo *responsive* (la página se adapta al tamaño de la pantalla)[8] y ofrecen estructuras muy fáciles de usar cuando se maqueta una página *web*. Va a facilitar el poder tener una estructura y una usabilidad ahorrando tiempo en el desarrollo.

En cuanto al alojamiento, se usará un servicio *Cloud* gratuito que es "Openshift"[9]. Un alojamiento en la nube (*Cloud*), ofrece más facilidades como la escalabilidad (si, por ejemplo, hay un pico de usuarios, se pueden usar más recursos de forma automática) y suele permitir desarrollar en diferentes lenguajes. En cambio, un *hosting* típico, tiene las limitaciones de los recursos y que, por norma general, solo permite que el *back-end* (o el código desarrollado en el servidor) sea solamente "PHP".

Por contra, la configuración de un servicio *Cloud* es mucho más complicada y no es un "*plug and play*" como lo es un *hosting* convencional, por tanto se requiere de unos conocimientos técnicos avanzados a nivel de gestión de sistemas. Además, se ha de realizar un estudio de los costes económicos que implican.

Para páginas *web* o aplicaciones sencillas que no requieran muchos recursos, la mayoría de servicios *Cloud* ofrecen soluciones gratuitas, mientras que es difícil obtener un alojamiento web normal con coste cero y que brinden un servicio correcto. Pensemos, además, que los planes gratuitos ofrecidos por una *Cloud* poseen unos recursos (memoria "RAM", uso de "CPU", almacenamiento en bases de datos) que siempre son mayores que la gran mayoría de los alojamientos normales y corrientes. Al final, la cuestión que decanta el uso de uno u otro, es la configuración y los conocimientos técnicos.

En la tabla a continuación, presentamos un pequeño resumen entre tres tipos de sistemas para alojar proyectos *web* que hay hoy en día (tabla realizada en base a los conocimientos del autor de este proyecto):

Servicio	**Precio**	**Lenguaje**	**Bases de datos**	**Hardware**	**Arquitectura alojada**	**Observaciones**
Hosting *web*	Pago	"PHP" casi siempre	"SQL"	Compartido y recursos muy limitados	Servidor y cliente	Fácil de gestionar
Cloud	*Freemium*	Libre elección entre varios lenguajes	Libre elección entre varias BDD's "NoSql" y "SQL"	Recursos elevados (son como máquinas virtuales)	Servidor y cliente	Conocimientos técnicos para configurar
MBaaS	*Freemium*. Cobro por usuarios / peticiones	"javaScript"	Agnóstico al usuario (transparente)	Agnóstico al usuario (transparente)	Solamente servidor. Necesita *hosting* donde alojar cliente	Actúa como "API" para los clientes

3. Objetivos generales

Se pueden definir y diferenciar los objetivos de la siguiente forma:

3.1 Objetivos principales

Objetivos de la aplicación:

- Rapidez de funcionamiento
- Sencillez de uso
- Usabilidad de la interfaz
- Accesibilidad desde cualquier dispositivo a partir (mínimo) de una tbleta
- Facilitar la gestión de un pequeño negocio

Objetivos para el usuario:

- Ahorro de costes al no depender de un *hardware*
- Registro de ventas
- Registro de clientes
- Contabilidad básica automática por empleados y fechas
- Acceder a datos desde cualquier lugar y con cualquier dispositivo, sin depender de un ordenador en concreto
- Tranquilidad en cuanto al almacenaje de los datos

Objetivos personales del autor:

- Estudio del desarrollo de una herramienta en la nube
- Estudio y mejora de la usabilidad de una aplicación ya existente en el mercado
- Mejora del desarrollo front-end: el proyecto (dado el tiempo disponible y a querer incidir en esta característica) va a ser prácticamente el diseño “HTML5-CSS3” y “javaScript” de la parte visual. Se va a incidir en las pantallas, en el flujo, en la experiencia de usuario y en que la curva de aprendizaje sea la más baja posible por parte de un usuario medio. Por lo tanto, va a girar todo en torno a lo que se ve. Se usarán herramientas on-line de *wireframes*, se estudiarán *frameworks*[9] “HTML5-CSS3” y *frameworks* “javaScript” que permitan realizar una web lo más atractiva posible. Aclarar, que el autor se dedica profesionalmente al desarrollo en “javaScript” (*front-end developer*) y puede aplicar sus conocimientos con los adquiridos en el máster

3.2 Objetivos secundarios

Objectivos adicionales que enriquecen el TF.

- Gestión de un proyecto
- Desarrollo completo desde las maquetas hasta el producto final
- Observar la diferencia entre planteamiento inicial y producto final

4. Metodología y proceso de trabajo

El fin de este proyecto es implementar un producto nuevo en base a los ya existentes. Es decir, estudiando la interfaz y la funcionalidad de productos existentes en el mercado, se quiere implementar un software más usable respecto a las soluciones actuales en el mercado.

No se han encontrado muchos "TPV" en la nube, ya que la mayoría son productos instalados en ordenadores locales, por lo que no son aprovechables las interfaces (que en web son "HTML5").

Básicamente se han usado las siguientes técnicas:

- *Kanban*: se han ido creando tareas pequeñas de diseño en "HTML5" y "CSS3" y de desarrollo en "javaScript". Estas tareas se iban escribiendo en pequeñas tarjetas (tal como el método indica) en tres columnas con el siguiente orden: diseño, programación y *testing/deployment*.
- Prototipos: se han creado prototipos con herramientas de prototipaje (en este caso las del servicio online lucidchart.com) y a partir de ahí, se han realizado las maquetas "HTML5-CSS3" y por último generar la lógica e interacciones con "javaScript".

A medida que se desarrollaba el código "HTML" se intentaba hacer un SEO correcto, rellenando ciertos atributos que pueden ayudar a un buen posicionamiento natural.

Por otra parte, el método de trabajo diario era desarrollo y subir a la nube (a "Openshift") los cambios implementados de forma asidua. De esta forma, ante cualquier imprevisto en local (perdida de datos, eliminar código por error, fallos en la máquina, etc…) se tenía una última copia subida perfectamente productiva.

5. Planificación

A continuación mostramos una tabla de hitos y un diagrama de Gantt con las tareas principales (*work-packages*) que se presuponen para el proyecto. En cuanto a los días dedicados, los fines de semana se han considerado laborables. También, se han marcado los días que (en principio) no se han podido dedicar al proyecto debido a asuntos personales y un viaje ya previsto del autor. Dado que habían muchas jornadas que no se han podido dedicar al proyecto (ya descritos en la tabla), en el cierre del proyecto (PAC 5) se ha seguido desarrollando la *web*.

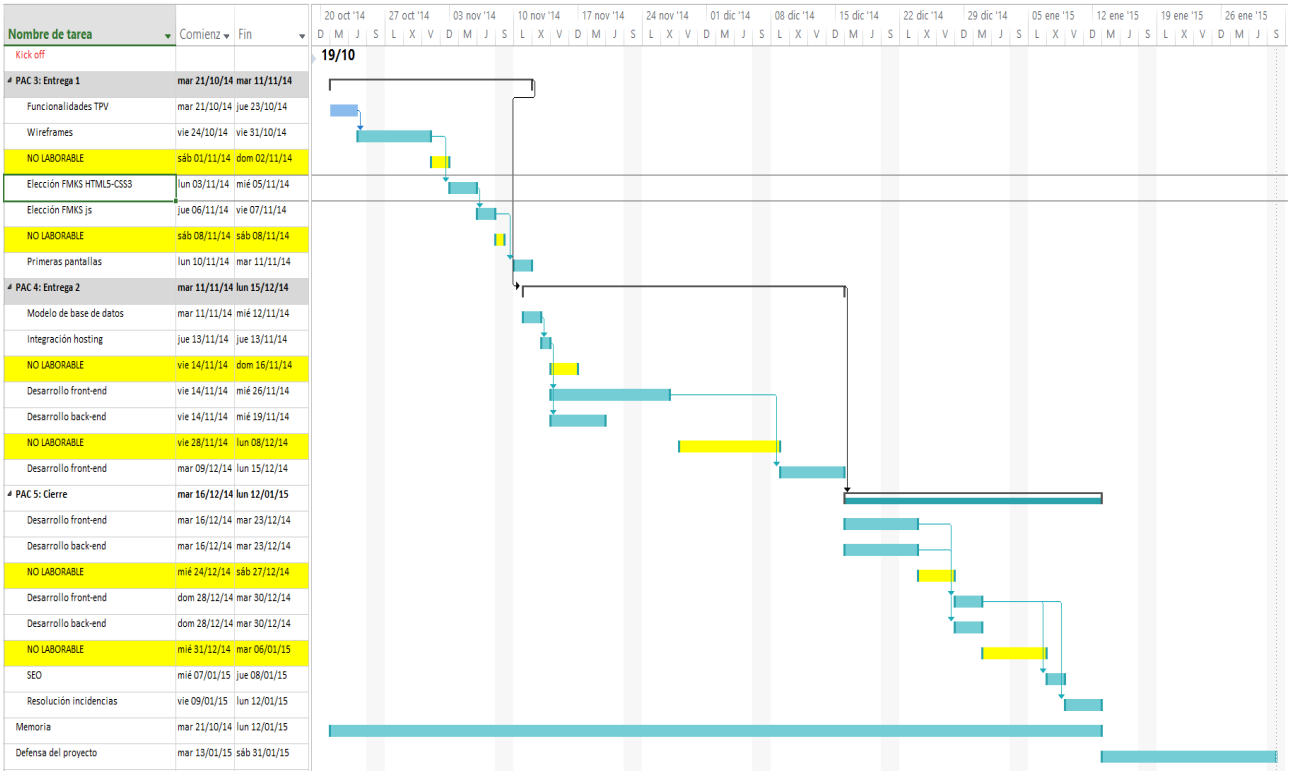
Sobre los riesgos, más allá de los imprevistos, se ha determinado que existirían probables retrasos en las entregas. En estos casos, el autor se ha puesto en contacto con el consultor de la asignatura y ha acordado una nueva fecha de entrega.

En la tabla siguiente se puede observar que se había previsto un desarrollo de lado servidor (back-end) pero finalmente no se realizó, y los esfuerzos se centraron en la parte front-end.

NOMBRE	DURACIÓN	INICIO	FINAL
PAC 3: Entrega 1	21 días	21/10/2014	10/11/2014
Días no dedicables al proyecto	4 días		
Días laborables que se pueden dedicar (no a jornada completa)	17 días		
Funcionalidades TPV	3 días	21/10/2014	23/10/2014
Wireframes	8 días	24/10/2014	31/10/2014
Elección/pruebas *frameworks HTML5-CSS3*	3 días	3/11/2014	5/11/2014
Elección/pruebas *frameworks JavaScript*	2 días	6/11/2014	7/11/2014
Primeras pantallas con los *frameworks* escogidos	2 días	9/11/2014	10/11/2014
PAC 4: Entrega 2	35 días	11/11/2014	15/12/2014
Días no dedicables al proyecto por viaje personal	14 días		
Días laborables que se pueden dedicar (no a jornada completa)	21 días		
Implementación base de datos	2 días	11/11/2014	12/11/2014
Integración *cloud hosting/MBaaS front-end*, base de datos y *back-end*	2 días	13/11/2014	14/11/2014
Desarrollo *front-end*	17 días	17/11/2014	15/12/2014
Desarrollo *back-end*	4 días	24/11/2014	27/11/2014
PAC 5: Cierre	28 días	16/11/2014	12/1/2015

Días no laborables no dedicables al proyecto		14 días		
Desarrollo *front-end*		10 días	16/11/2014	12/1/2014
Desarrollo *back-end*		5 días	16/11/2014	12/1/2014
SEO		2 días	8/01/2015	9/01/2015
Resolución incidencias		10 días	16/11/2014	12/1/2015
Memoria	84 días	21/10/2014	12/1/2015	
Defensa del proyecto	17 días	13/1/2015	30/1/2015	

El diagrama de Gantt del proyecto es el siguiente:



6. Estructura del resto del documento

- Capítulo 2: Las soluciones cloud que hay en la actualidad. Sus pros y contras. Público objetivo y clientes potenciales
- Capítulo 3: Diseño de la aplicación. Arquitectura tecnológica. Diagramas de casos de uso, d estado y *mindmap*. Estilos empleados. Estudio de la usabilidad y explicación del diseño responsive de la solución. ¿Qué lenguajes de programación se han usado?
- Capítulo 4: Instalación y plataformas y navegadores en las que se ha probado con éxito "TPVCloud".
- Capítulo 5: Enlace a vídeo demostrativo del uso. Enlace a la aplicación *web* y usuarios con contraseñas. Enlace a los prototipos (maquetas) realizadas antes del desarrollo.
- Capítulo 6: Conclusiones con las lecciones aprendidas. Líneas de futuro con ampliaciones y mejoras.

Capítulo 2: Análisis

1. Estado del arte

"Internet vino para quedarse": esta afirmación ahora mismo es anacrónica. Pero no está fuera de lugar afirmar que está en fase de "expansión" el desarrollo de *webs* que sustituyen al *software* de escritorio (instalado en un ordenador) vía *La Nube* (*Cloud*). ¿Y qué es *La Nube*? "Amazon"[9] la define como "la entrega de recursos de T.I. a través de Internet". Excelente explicación.

Y ya que *La Nube* está en pleno período de expansión e implantación, surgen lenguajes de programación que existen hace años pero que ahora están teniendo, también, su lanzamiento a nivel mundial, su fervor por las masas "*nerds*" y su reconocimiento. De igual manera, ahora se da una importancia nunca antes conocida a la experiencia de usuario, a poder usar *webs*, *apps* y *software* en general con una interfaz elegante y sencilla a la par que brinde una interacción entre humano y dispositivo de lo más agradable, acercando el uso de la computación en general a cualquier persona independientemente de su nivel cultural.

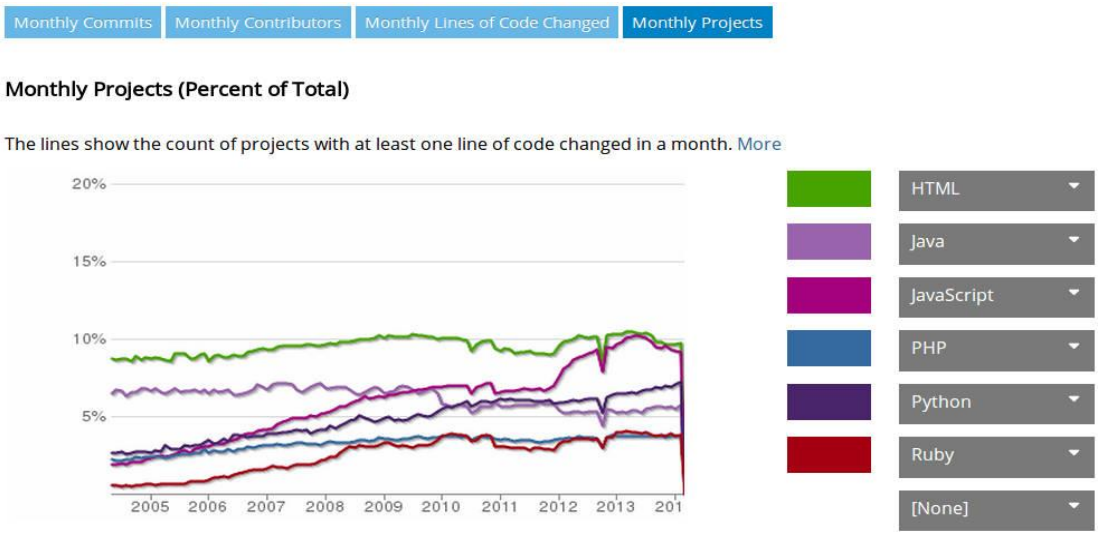
Hasta no hace mucho, programar una *web* era básicamente en lenguajes como "PHP*"* (a lo sumo "JAVA*"* o ".NET*"* de "Microsoft*"*) en el lado servidor y "HTML4*"* con algo de "*javaScript"* y la ayuda de *"*Flash*"* para realizar animaciones.

Pero la implantación del potente "HTML5" y "CSS3" ha permitido un giro importante en el desarrollo *web*, sobre todo por el acercamiento hacia una *web* más semántica y la incorporación de elementos para "jugar" con las animaciones y todo lo relacionado con lo multimedia (adiós "Flash" y "Silverlight"). Asimismo, "javaScript*"* (otrora vilipendiado por la mayoría de la comunidad de programadores) renace como un claro competidor respecto a los grandes dinosaurios ("JAVA*"*, "PHP*"* y ".NET*"*) en Internet, dada su orientación (desde sus inicios) al desarrollo *web*. Éste es un lenguaje dinámico, es decir, no se compila[10], como la mayoría de los lenguajes que están ahora mismo siendo más usados.

Y dentro de esta jauría de lenguajes (dinámicos) que están creciendo a la sombra del desarrollo *web*, se sabe que los que mayor crecimiento están teniendo (a nivel de proyectos y de comunidad) están establecidos en esta tabla[11]:

Dynamic Language	Growth Rank
Python	1
Javascript	2
PHP	3
Ruby	4
Perl	5
Tcl	6

"Python" y "javaScript" están en las dos primeras posiciones de lenguajes dinámicos que están creciendo con una mayor velocidad. Para corroborar aún más éstos datos, en el siguiente gráfico (se puede acceder y cambiar datos desde [12]) se comparan con lenguajes "de siempre" como "JAVA" o "PHP" y observamos como "los de siempre" se estancan y los nuevos están en fase de expansión:



Indudablemente, el fin de este proyecto es realizar un i+d profundo de alguno de estos lenguajes que se están imponiendo y conocer sus virtudes y defectos, jugar con sus secretos y saber cuál es el que más ofrece una curva de aprendizaje menos dura con un resultado más exitoso. Y, por supuesto, dar una vuelta de tuerca a la experiencia de usuario del *software* que se desarrollará, pues visto lo que hay en el mercado en lo que se refiere a *TPV's*, se puede afirmar que no ha habido mucho interés en mejorar interfaces que bien parecen de hace diez o quince años.

En el lado del cliente también hay una guerra abierta entre diferentes empresas que ofrecen ya sus propios frameworks "HTML5–CSS3" para hacer páginas web (o *webapps*) de una forma fácil, agradable y que se puedan adaptar a diversos dispositivos. Facilitando la vida al desarrollador dejándole más tiempo para la programación e interacción entre cliente y servidor. En el caso del presente proyecto, como se ha nombrado antes, se hará uso del *framework* son *Bootstrap*.

En lo que respecta al proyecto, un terminal punto de venta, es un programa usado en millones de establecimientos alrededor del mundo. Es, simplemente, una caja registradora informatizada y con un una interacción más amable que los botones de la misma caja.

Dada la importancia de esta función en cualquier pequeña y mediana empresa (PYME), a día de hoy, siguen habiendo multitud de negocios con una caja registradora o un *software* instalado en ordenadores propios.

¿Qué problemas se evidencian? Los de las cajas registradoras son evidentes si alguna vez se han usado:

- Configuraciones imposibles de ejecutar: por mucho que una menta privilegiada se lea de cabo a rabo el manual de instrucciones, tareas sencillas como cambiar el IVA o el número de empleados puede requerir de ocho licenciaturas y tres doctorados. ¡Imaginemos al señor peluquero o de la lampistería! (sin faltar al respeto a la inteligencia de nadie, claro está).

- Sin comunicaciones: ¿Y si se desea enviar un email con la contabilidad diaria a un gestor? ¿Y si se desea acceder desde casa a ver qué está ocurriendo con las ventas diarias?.
- *Hardware*: ¿Qué ocurre si repentinamente la máquina deja de funcionar? ¿Qué fue del registro de ventas diarias, semanas o mensuales? Se fueron al más allá igual que la máquina.

Y ahora pasemos a ver las contras de los *software* TPV instalados en ordenadores:

- Caros: requieren licencias anuales y mensuales de un coste elevado.
- *Hardware* a medida: la mayoría vienen con cajones de monedas y billetes y pantallas táctiles. Muchas empresas se aprovechan de esto para vender "*packs*" de *software* y *hardware* por más de mil euros.
- Acceso a los datos desde otros dispositivos: dependiendo del *software* se podrá acceder, pero acceder desde un ordenador a otro vía internet, no está al alcance del usuario medio.
- Pérdida de datos: se está en lo de antes con la máquina registradora. ¿Y si deja de funcionar el ordenador? Se puede acudir a una empresa que recupere datos con el elevado coste que ello implica. O se pueden usar otras soluciones como intentar hacer copias de seguridad a menudo, pero se sabe que el usuario medio no hará semejante acción muy seguido.

En fin, que hay un panorama nada halagüeño en lo referente, sobre todo, a la perdida de datos y al acceso desde cualquier parte a los mismos.

¿En cuanto a la interfaz de la mayoría de estos *softwares*? La siguiente pantalla habla por sí sola. Si cualquier persona repara durante unos segundos en la pantalla de alguna TPV de cualquier comercio a pie de calle, se dará cuenta que no dista mucho de la misma:

2. Análisis del mercado

En cuanto a los servicios actuales de TPV en *La Nube*, una simple búsqueda de las palabras "tpv online" da unos resultados (a entender del autor) algo esperpénticos. En la primera página de resultados (con unos mínimos conocimientos de *SEO-SEM* se sabe que la gran mayoría no pasa a la segunda página) son de servicios para tiendas *on-line* de entidades bancarias, es decir, aplicaciones a acoplar a gente que tenga una tienda en Internet (¿y para los que no?). Hay un par de servicios que sí que se adecúan a lo que se busca (*onlinetpv*.com y *gesio.com*) con precios dispares e interfaces no muy diferentes de la pantalla presentada anteriormente.

Ya si se realiza una búsqueda en inglés como "*POS online*" encontramos mayores resultados y productos de mayor calidad: *vendhq.com*, *shopify.com/pos* e *imonggo.com*.

Respecto a estas diferencias, el autor de este proyecto no entiende qué hay de diferente entre el mundo anglosajón y el hispano para realizar proyectos tan diferentes y con tanta distancia de calidad (visual) entre aplicaciones realizadas en ambos mundos. Por lo tanto, el objetivo es poder realizar una TPV *online* accesible a todo el mundo en esta parte del planeta, con la calidad (al menos visual) de las que se hacen en otros países.

2.1 Estudio de mercado

La audiencia potencial de un software como el de este proyecto es realmente amplio. Según el informe "Retrato de la Pyme" [13] de 2014 del Ministerio de Industria, Energía y Turismo español, habían alrededor de tres millones de PYMES y de micro empresas sin asalariados. Por lo tanto, se supone que un amplio número de estas empresas venderán servicios y necesitarán algún programa para cobrar y llevar una contabilidad mínima. De estas PYMES, alrededor de ochocientas mil empresas son de comercio, que sería el objetivo prioritario de un *software* como el desarrollado. Esto, solamente en España. Si se extrapola a otros países industrializados con una alta penetración de Internet, el número de potenciales usuarios crece exponencialmente.

Pensemos en cualquier pequeña ferretería, bar, tienda de cualquier índole… Y pensemos la cantidad de veces que hemos visto pequeños negocios con una simple caja registradora (a día de hoy) y sin ningún ordenador para llevar el registro de cuentas. Pero seguramente, esos empleados sí que tienen la mayoría un teléfono con Internet o una tableta que les permite acceder a Internet.

2.2 Comparativa de la competencia

NOMBRE	URL	PROS	CONTRAS
VendHQ	vendhq.com	· Excelente interfaz · Aplicaciones nativas para dispositivos móviles y *web* para ordenadores sobremesa · Modo *offline* · Conexión con diferentes tipos de pago en servicios *online*	· A partir de sesenta dólares al mes para quinientos productos activos · No queda clara la diferencia entre la web y la aplicación nativa
Shopify POS	shopify.com/pos	· Diferentes tipos de pagos con servicios online · Analíticas · Modo *offline* · Buena interfaz · Soporte 24 / 7	· Funciona solamente con el sistema Shopify (plataforma para crear una tienda online) · Necesario iPad · A partir de quince dólares al mes para solamente 25 productos
iMonggo	imonggo.com	· Gratuito hasta mil productos y mil transacciones mensuales · Modo *offline* · Modo de premios a buenos clientes	· Aplicación nativa para tabletas y móviles · No es una página *web*
POS tablet	postablet.com	· Proveen de equipo completo de hardware · Control absoluto de todo el proceso desde el inventario hasta la venta · Contabilidad avanzada · Interfaz clara y usable	· Necesario iPad · Plan de precios no es público (se supone que no es barato)

2.3 Estrategia de marketing

Después de la tabla anterior, este proyecto se debería enfocar en cubrir las debilidades de la competencia, que básicamente, son que o son aplicaciones nativas de dispositivos móviles (no una web accesible desde cualquier dispositivo) o su alto precio. Además la mayoría, a entender del autor, dan demasiadas características disponibles las cuales seguramente no las aprovechará una gran mayoría de usuarios.

Por lo tanto, se debería ir hacia una venta del producto *freemium*. Ofrecer las características básicas (sin un límite bajo de productos o transacciones) de forma gratuita y funciones avanzadas (como analíticas) por un precio mensual. Además, se debe vender el hecho de que esta aplicación web no está ligada a ninguna plataforma, ya sea un sistema operativo (“iOS” o “Android”) ni genérica para venta online (“Prestashop”, “Shopify”, etc…).

3. Público objetivo y perfiles de usuario

Un perfil de usuario así como el público objetivo vendría determinado por las características:

- Micro empresas de uno a diez empleados con servicio de venta al público en general
- Cualquier sector es objetivo, tendiendo más hacia el comercio o restauración
- Usuarios con cultura tecnológica que desean acceder a sus datos en cualquier momento
- Usuarios de cualquier nivel de uso de programas

4. Definición de objetivos/especificaciones del producto

Una TPV (terminal punto de venta) tiene la principal funcionalidad de registrar las ventas para un vendedor o comprador. Esta es su principal acción.

A partir de aquí podemos extender hasta donde queramos lo que debería hacer este sistema de registro de ventas. Podríamos simplemente tener un sistema que registra una venta y punto o llegar a sistemas complejos con gestión de clientes, inventarios, etc. Pero cuanto más extendamos sus funcionalidades podemos llegar a sistemas que ya existen en el mercado como puede ser un ERP o un CRM. Por lo que se ha de tener cuidado en hacer lo justo y necesario para que nos movamos en un entorno que sepamos que se está usando una TPV.

Así que partamos de un registro de ventas, a partir de aquí podemos tener:

- Registro de ventas de productos/servicios registrados en el sistema
- Registro de ventas de productos/servicios libres (de productos no registrados en el sistema)
- Gestión de inventario de productos vendidos
- Gestión de clientes
- Contabilidad básica de ventas (por fecha, tipo de cobro, por cliente, etc…)
- Gestión de vendedores
- Estadísticas de ventas

A grandes rasgos estas son las principales funcionalidades, que si queremos generalizarlas podemos realizar los siguientes grupos:

- Registros ventas
- Inventario productos/servicios
- Gestión clientes
- Contabilidad/estadísticas
- Gestión vendedores

Otros objetivos son:

- Modelo responsive: mínimo accesible desde una tableta
- No depender de un sistema operativo ni una plataforma de tiendas online
- Usabilidad fácil (fácil para gente reacia al uso de este tipo de programas)

El producto final contiene las siguientes funcionalidades. Se muestra con un asterisco las funcionalidades que no se han podido desarrollar finalmente y que se expondrán en un capítulo siguiente:

- TPV:
    - Calculadora completa para registrar líneas de venta de productos no registrados
    - Agregar líneas de ventas con productos existentes
    - Escoger un producto según la categoría*

- Listado de productos:
    - Listar productos
    - Crear un nuevo producto*
    - Borrar un producto
    - Editar la información de un producto
    - Listar categorías*
    - Editar / Crear / Eliminar categorías*

- Listado de clientes:
    - Listar clientes
    - Crear un nuevo cliente*
    - Editar información de un cliente
    - Borrar un cliente

- Administración:
    - Estadísticas por ventas totales
    - Estadísticas por ventas de empleados
    - Estadísticas por ventas de fechas
    - Datos de la compañía*
    - Listado empleados
    - Editar información empleados*

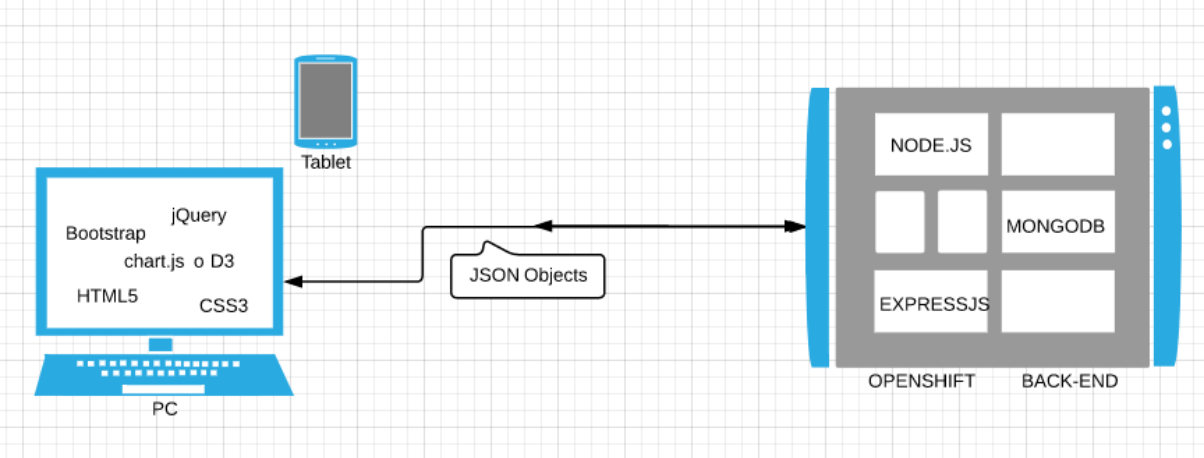
Es importante remarcar que toda la información que se muestra y se modifica, no se contemplará cuando se recargue la página, pues no hay una base de datos que la almacene (no hay persistencia de datos).

Capítulo 3: Diseño

1. Arquitectura general de la aplicación

Al ser una aplicación *web*, se establece un modelo “on line”. El significado de esto es que se deberá tener una conexión a internet para poder usar el sistema. No se descarta que en un futuro se pueda disponer de una “app” para instalar en escritorio o en un dispositivo móvil, pero, de momento, queda fuera del alcance de este proyecto.

Las posibles desventajas de este modelo, es que el usuario no disponga en algún momento de conexión a Internet. Bien, en este caso (suponiendo que está haciendo uso del sistema), se guardarán los datos en local, haciendo uso de la funcionalidad “Local Storage” de los navegadores actuales. Esta es una pequeña base de datos que implementan los “browsers” para persistencia de datos aunque se apague el ordenador. Por lo que el sistema proveerá de una lógica para tener en cuenta si hay datos no salvados “on line” y actualizar con los que quedaran “off line”. Se ha de tener especial cuidado en este aspecto, pues tratamos con contabilidad y se podría incurrir en inconsistencias y, por tanto, incluso poder hacer que el usuario tuviera problemas incluso legales por no tener una contabilidad clara y concisa.

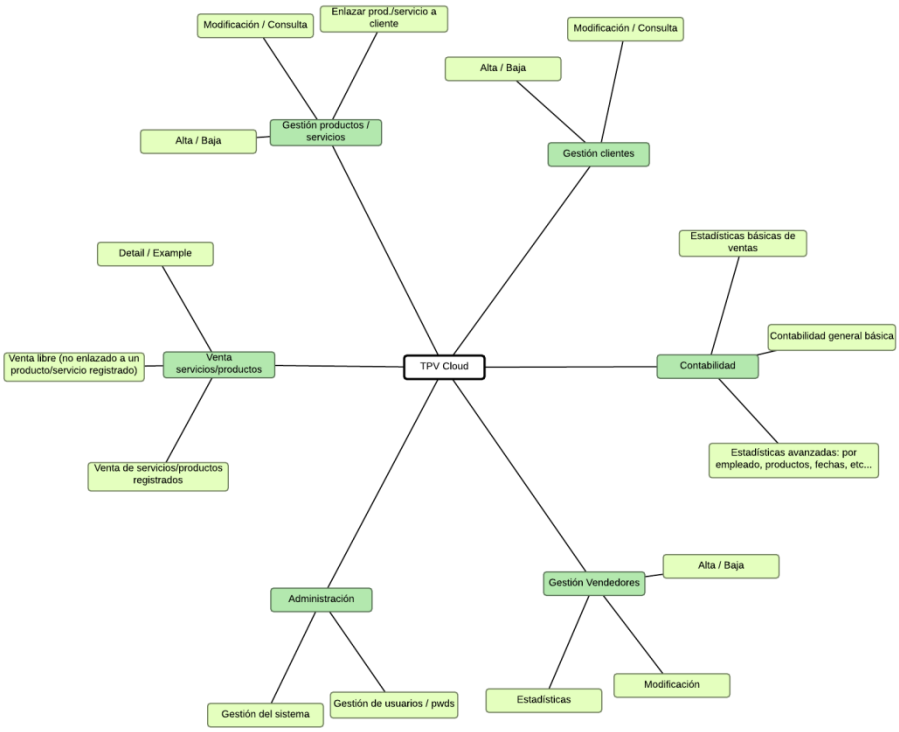


Tal como observamos en el diagrama anterior la relación es la siguiente:

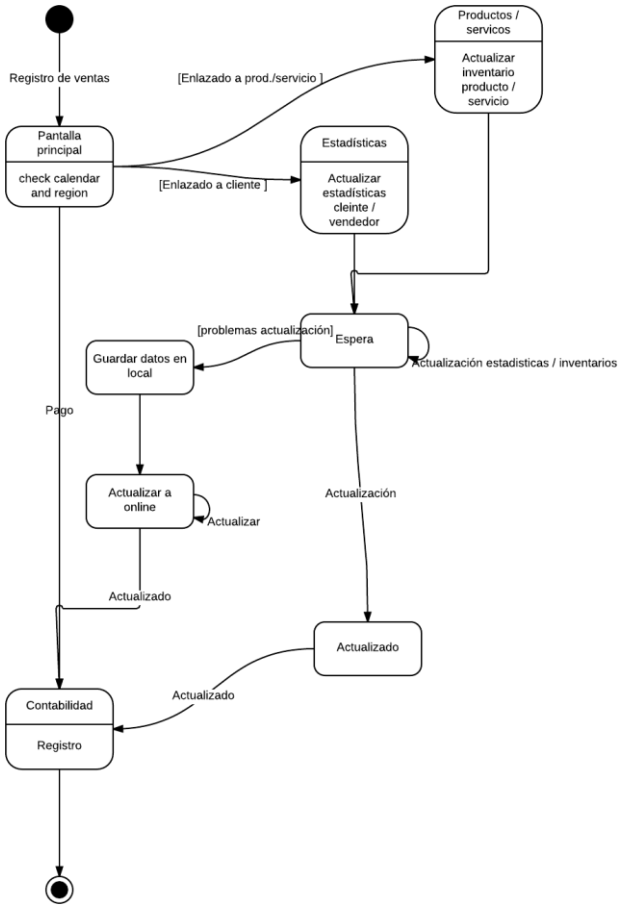
- Tanto los ficheros “HTML5”, “CSS3” y “javaScript” están alojados en “Openshift”, servicio *cloud* de “Red Hat” con un buen servicio gratuito. En el servidor “Openshift” ahora mismo hay alojado un pequeño servidor desarrollado en “Node.js” con una base de datos “MongoDB”. Como se explicó antes, apenas se ha desarrollado nada y la información está guardad en los ficheros “javaScript” que se le envía al cliente.
- El cliente cargará los archivos “HTML5”, “CSS3” y “javaScript” así como librerías como “Boostrap” o “chart.js”.
- El intercambio de información (en el momento en que la base de datos se implemente) será con objetos “JSON”.

2. Arquitectura de la información y diagramas de navegación

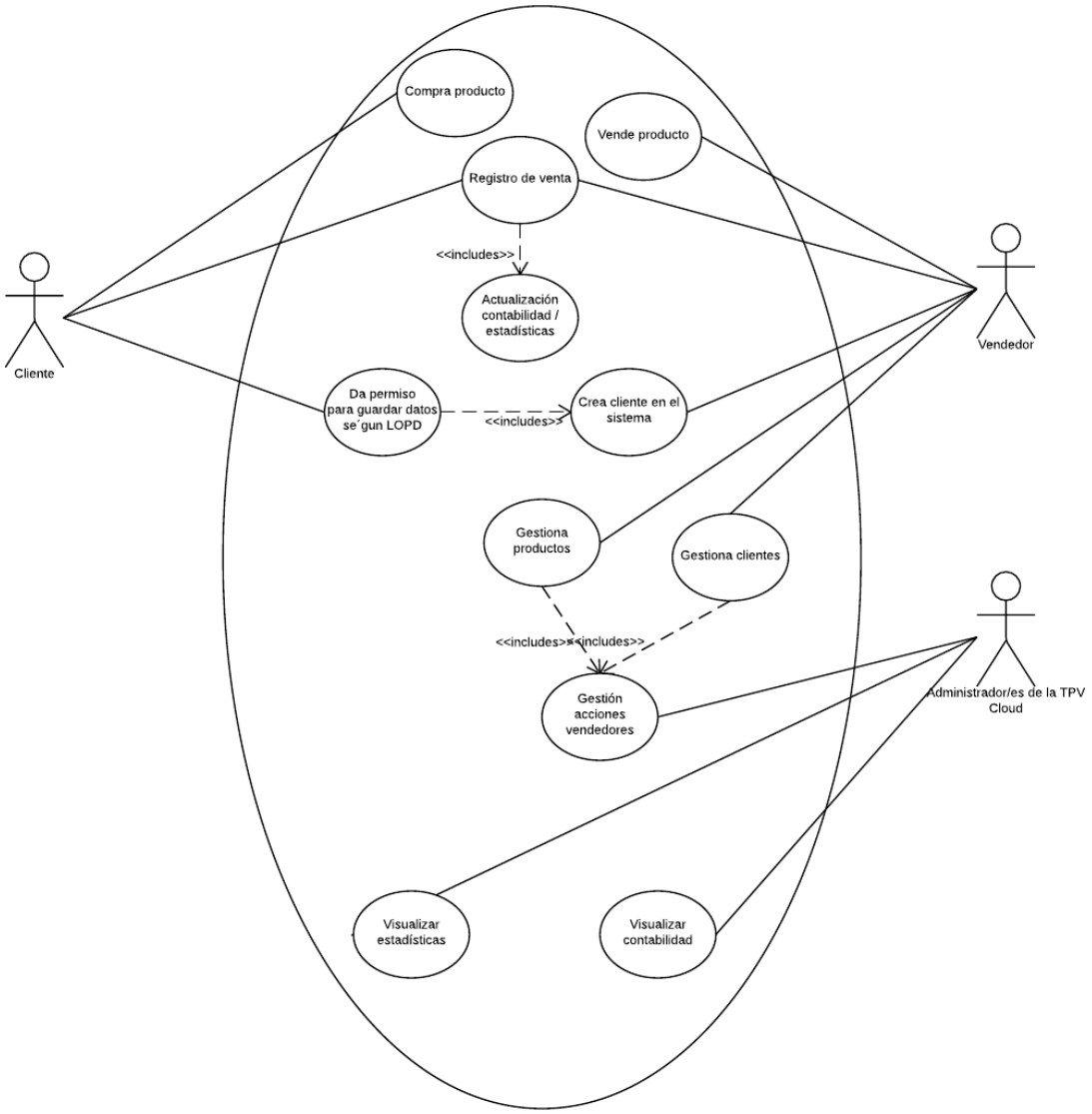
En el siguiente diagrama, podemos observar un *mind map* de las funcionalidades que ayudan a establecer un primer pasa para la navegación y estructura de la aplicación:



El diagrama de estados de una venta típica sería la siguiente:



Y un esquema de casos de uso lo encontramos tal que:



3. Diseño gráfico e interfaces

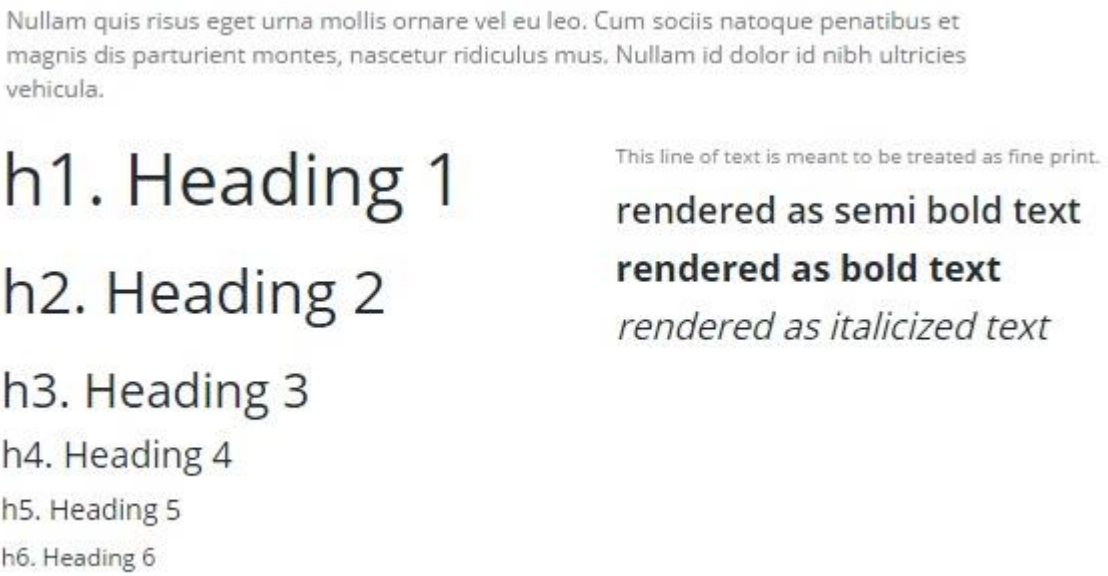
En cuanto a las interfaces, a su lenguaje, cabe decir que se ha realizado el proyecto en inglés por más comodidad del autor, que suele trabajar en su vida laboral (a la hora de desarrollar) en ese idioma.

3.1 Estilos

- Logotipo:

- Paleta de colores:

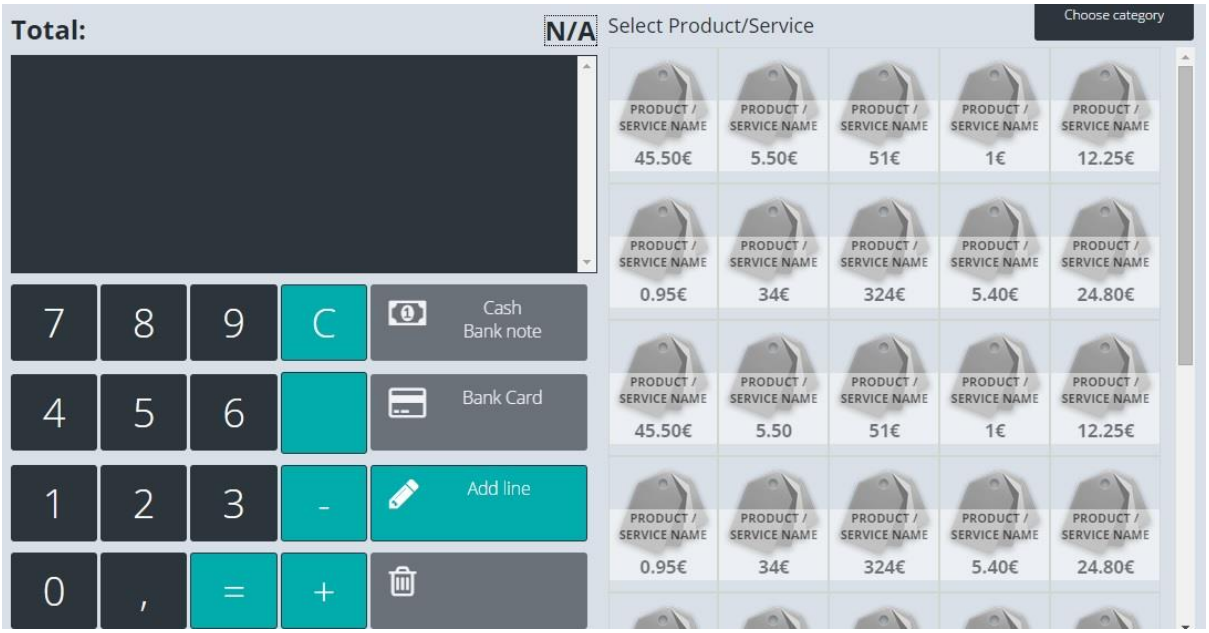
- Paleta tipográfica y estilo de fuentes:
  La fuente que se ha empleado ha sido una “Open Sans” jugando con diferentes tamaños. El tamaño estándar es de 12px . A partir de ahí se ha jugado con las medidas “em” [14]



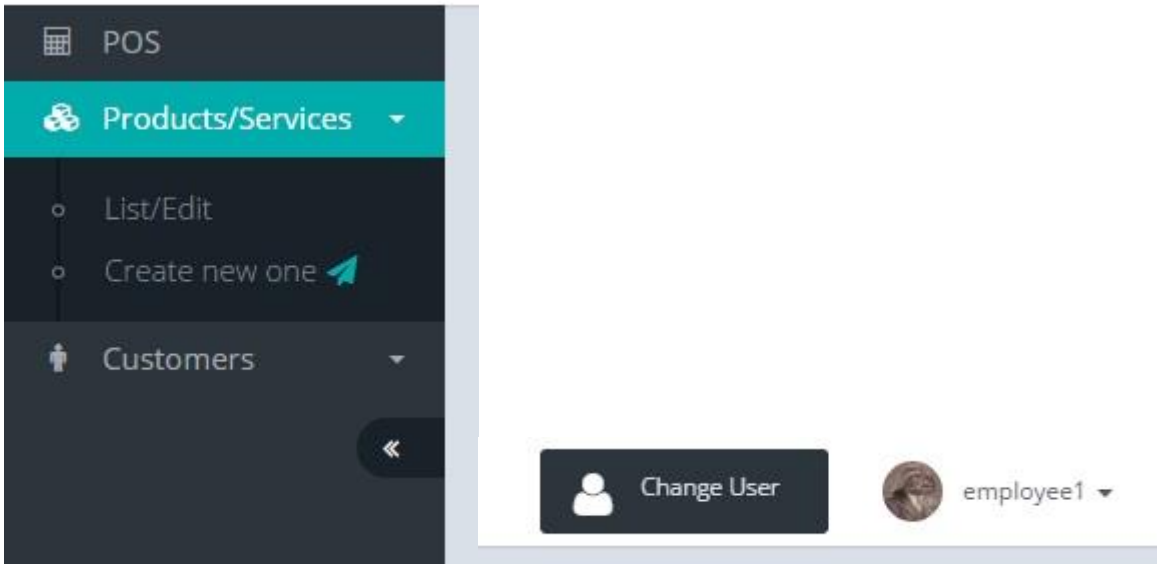
- Botones e iconos:
  Se han usado los iconos de la fuenta “Font awesome icons” (http://fortawesome.github.io/Font-Awesome/icons/) que contiene una extensa librería de iconos para diseño web. Y se han usado los siguientes botones:

3.2 Usabilidad/UX

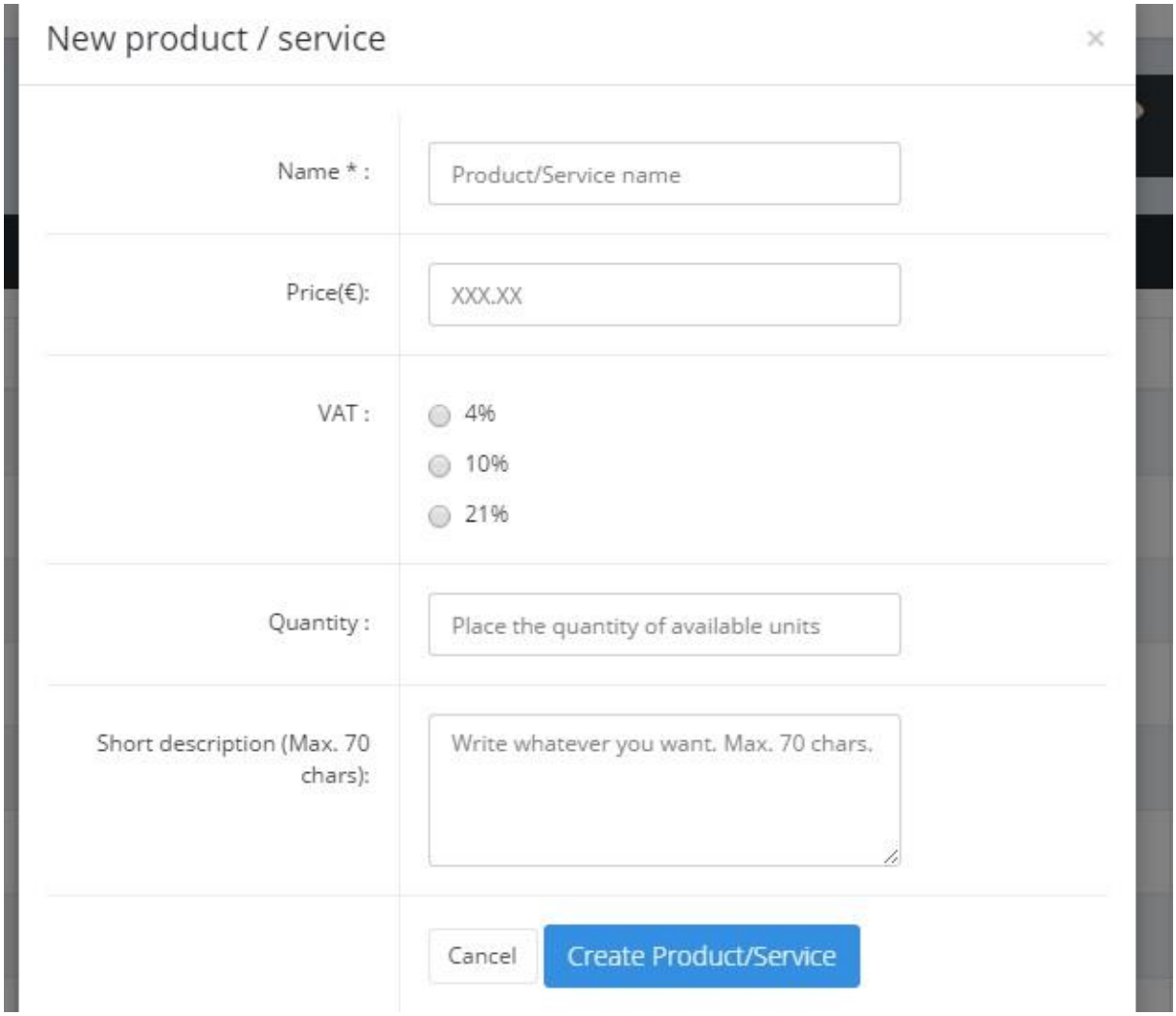
En el apartado de la usabilidad se ha estudiado que la pantalla principal (calculadora y selección de productos para añadir a la venta actual) fueran botones grandes para que sea fácilmente usable desde una pantalla táctil:



Hay dos menús principales, uno superior con la información del usuario actual y un botón para cambiar de usuario. No se pide contraseña, pues no facilitaría el uso de la TPV si hay varios empleados que han de cobrar y anotar la venta a ellos mismos. El menú de cambio de sección (productos, clientes y administración) se ha puesto en la parte de la izquierda, el cual se puede esconder para ganar espacio:

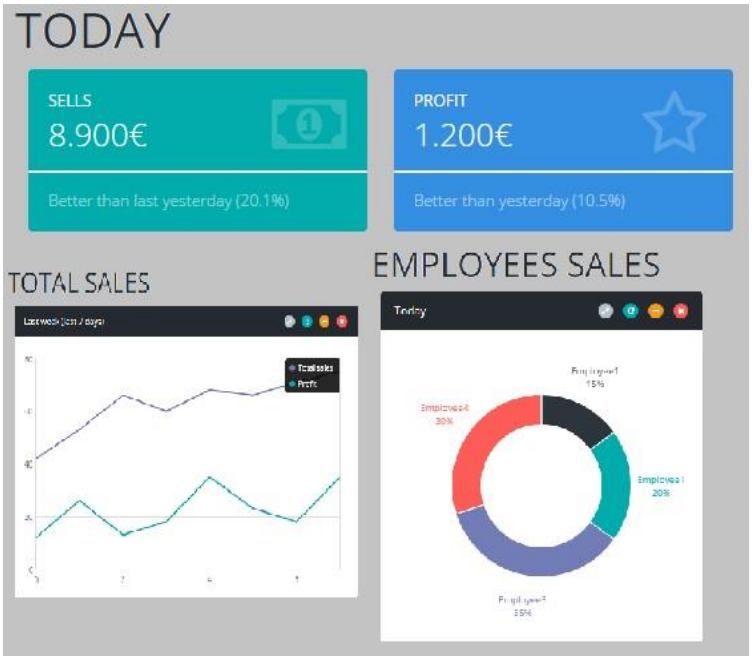


Por otra parte, en los listados de productos y clientes, cualquier interacción para ver el detalle de una entidad (o crear) se hace desde una modal y no cambiando de pantalla, ganando fluidez:



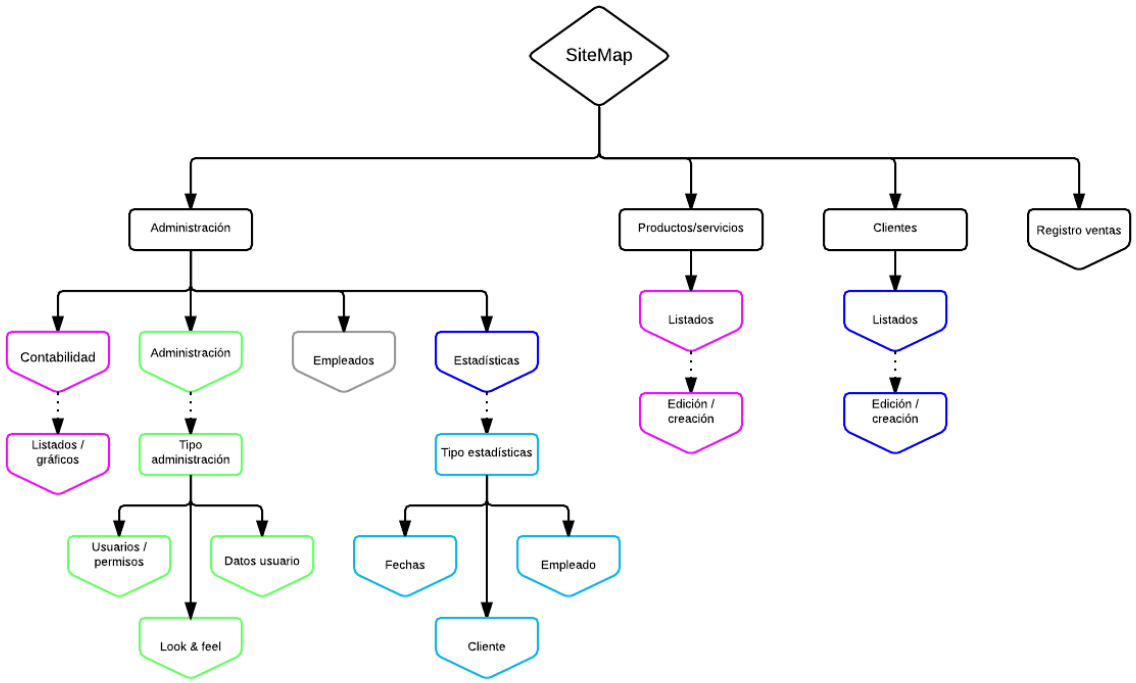
:

En cuanto a las estadísticas, se ha buscado simplificar con gráficos grandes y claros:



Es muy importante recalcar que cuando se cambia de sección o aparece cualquier modal, solamente se recarga el elemento central de la aplicación, no toda la página como hacen la mayoría. Con esto ganamos rapidez y que la sensación del usuario sea de que tiene ante sí un sistema con mucha fluidez. Así, si cambiamos de sección y volvemos a la TPV y si estábamos en medio de una venta, no perdemos esa acción y continúa donde estaba.

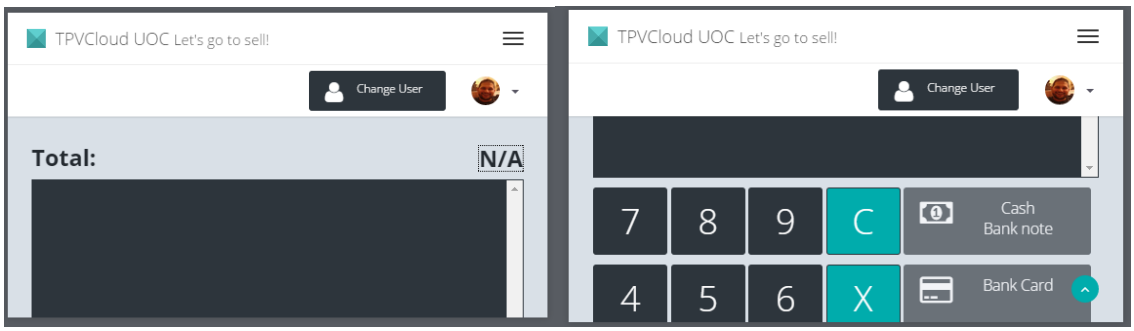
Por último mostrar el mapa de navegación:



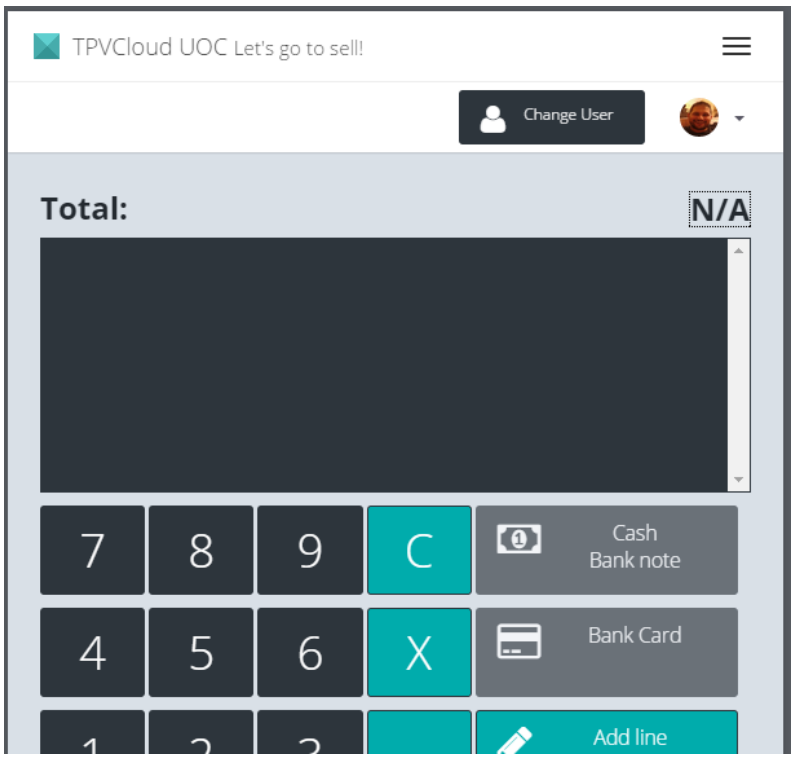
3.3 Diseño responsive

El diseño funciona correctamente para pantallas en las siguientes resoluciones:

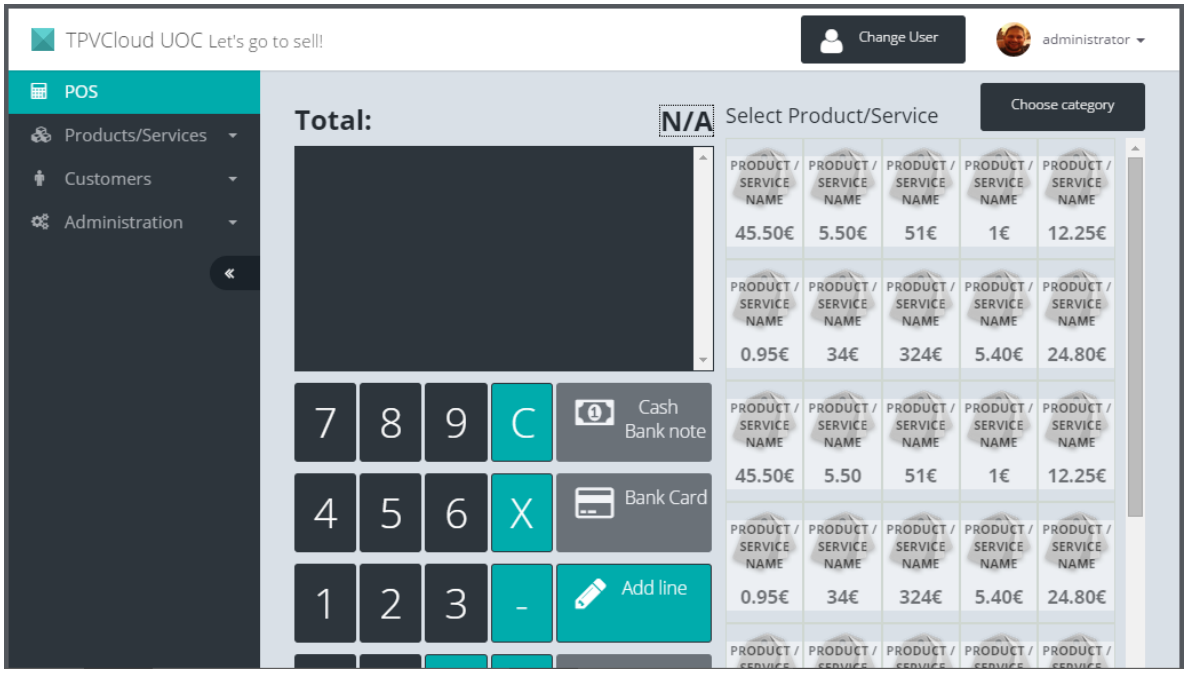
- 558 x 320 px (ej.: iPhone con la pantalla en horizontal o apaisada)



- 600 x 800 px (ej.: tabletas antiguas pequeñas)



- 1024 x 768 px (ej.: tabletas modernas como iPad 3)



4. Lenguajes de programación

Necesitamos escoger un hosting *cloud*. No vamos a escoger un alojamiento web normal y corriente que puede ofrecer cualquier compañía que podamos encontrar, pues la mayoría solamente aceptan **"**PHP**"**. Además, necesitamos control para instalar la base de datos que escojamos, ver archivos logs, etc... que un alojamiento normal no puede brindar. Una ventaja de desarrollo en un *hosting Cloud* es la escalabilidad. Podemos configurarla para que si tenemos un pico de usuarios, automáticamente aumente el número de recursos disponibles. En nuestro caso no será aplicable, ya que será para un usuario y no prevemos la necesidad de grandes recursos. En *Cloud* para el desarrollo web encontramos dos tipos: *IaaS* y *PaaS* **:**

- *IaaS*
  Infraestructure as a Service. Básicamente es como poseer una máquina virtual en la que nos tenemos que hacer cargo de la gestión de casi todo. Por una parte necesitamos amplios conocimientos técnicos, pero podemos llegar a configurar el alojamiento a nuestro antojo, pudiendo resolver problemas de velocidad, espacio, memoria, etc... teniendo los conocimientos técnicos adecuados. Un ejemplo de esta infraestructura es **"**Amazon Web Services**"** (http://aws.amazon.org).

- *PaaS*
  Platform as a Service. Ofrece un modelo en el cual, no podemos llegar a modificar aspectos técnicos como en un *IaaS*. Es mucho más sencillo de usar y de instalar nuestro desarrollo. La mayoría ofrecen un panel de control fácil de usar e interacción con clientes instalados en nuestras máquinas. Nos desvinculamos de los conocimientos técnicos y podemos centrarnos exclusivamente en el desarrollo. Ejemplos son **"**Google App Engine**"** (http://cloud.google.com/appengine), **"**Openshift by RedHat**"** (http://openshift.com) y **"**Rackspace**"** (http://rackspace.com).

Para centrarnos en el proceso de desarrollo, escogeremos un *PaaS*, ya que no necesitamos afinar perfectamente nuestro servidor para miles de peticiones o usuarios. Simplificará el proyecto y podremos centrarnos en el desarrollo. Buscamos la simplicidad en cuanto al alojamiento. Así que analizando algunos *PaaS* se tiene que "Google App Engine" no ofrece servicio para Node.js (lenguaje de servidor que en el que se querrá desarrollar en el futuro), "RackSpace" es un servicio demasiado profesional, "AppFog" no ofrece un servicio gratuito (solamente una prueba de 30 días) y "OpenShift by Redhat" ofrece un nivel gratuito, es compatible con Node.js y varias bases de datos tanto relacionales como no. Además posee una documentación amplia y está basado en "OpenStack", que es un *IaaS* de código abierto usado por empresas como VMWare, AMD, HP e IBM.

Hechas las pruebas pertinentes, se ha comprobado que el funcionamiento es relativamente sencillo. Hay que instalar un cliente en el ordenador local y, mediante consola, podemos crear proyectos, subir cambios y hacer cualquier tipo de procesos con una curva de aprendizaje realmente baja. Todo está basado en un servidor GIT, un sistema de control de versiones ampliamente usado y de código abierto.

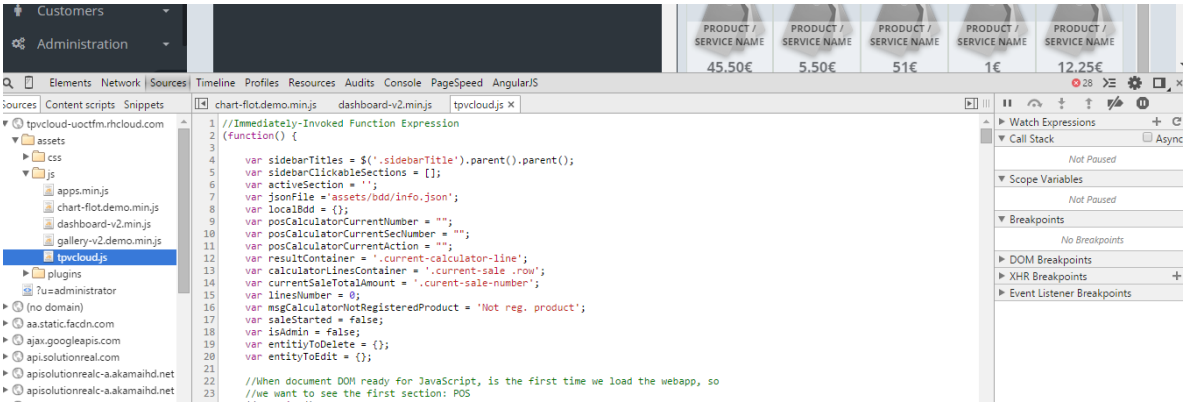
Por lo tanto, el *hosting cloud* será **"**Openshift by Redhat**"**.

Los lenguajes de desarrollo serán:

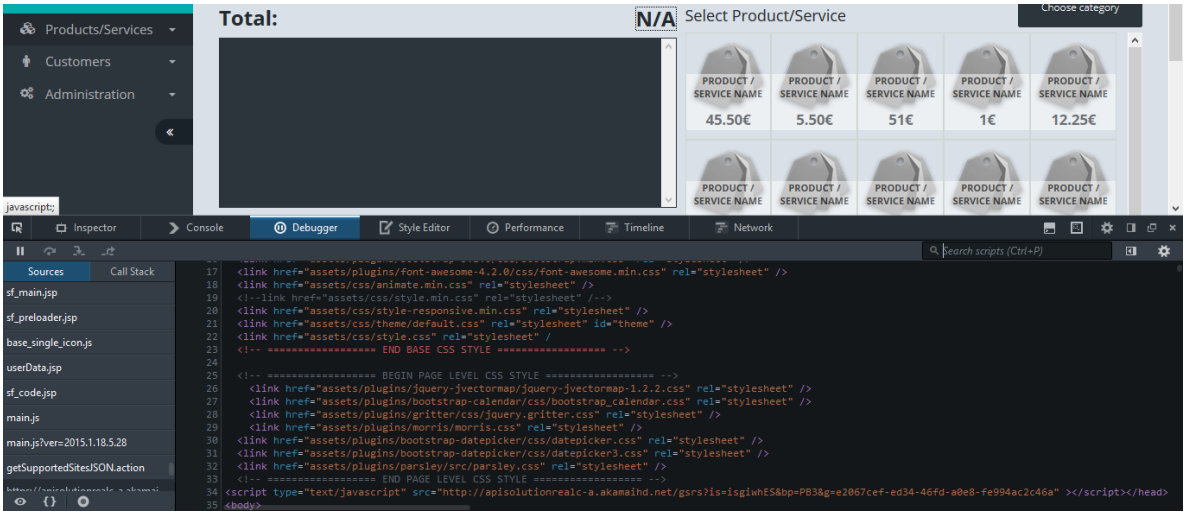
- "HTML5" y "CSS3": para aprovechas lo más novedoso a nivel de maquetación web, aprovechando el almacenamiento de los navegadores y nuevas funcionalidades haciendo una web más semántica. Se ha usado el framework Boostrap 3 que facilita el desarrollo en modo *responsive*.
- "javaScript": lenguaje por antonomasia usado en el desarrollo front-end. De este se han usado las siguientes librerías:
  - "jQuery"[15]
  - "chart.js": para implementar los gráficos de las estadísticas[16]
  - "modernizr.js": para asegurar retro compatibilidad con navegadores más antiguos[17]

Como herramientas de desarrollo se ha usado:

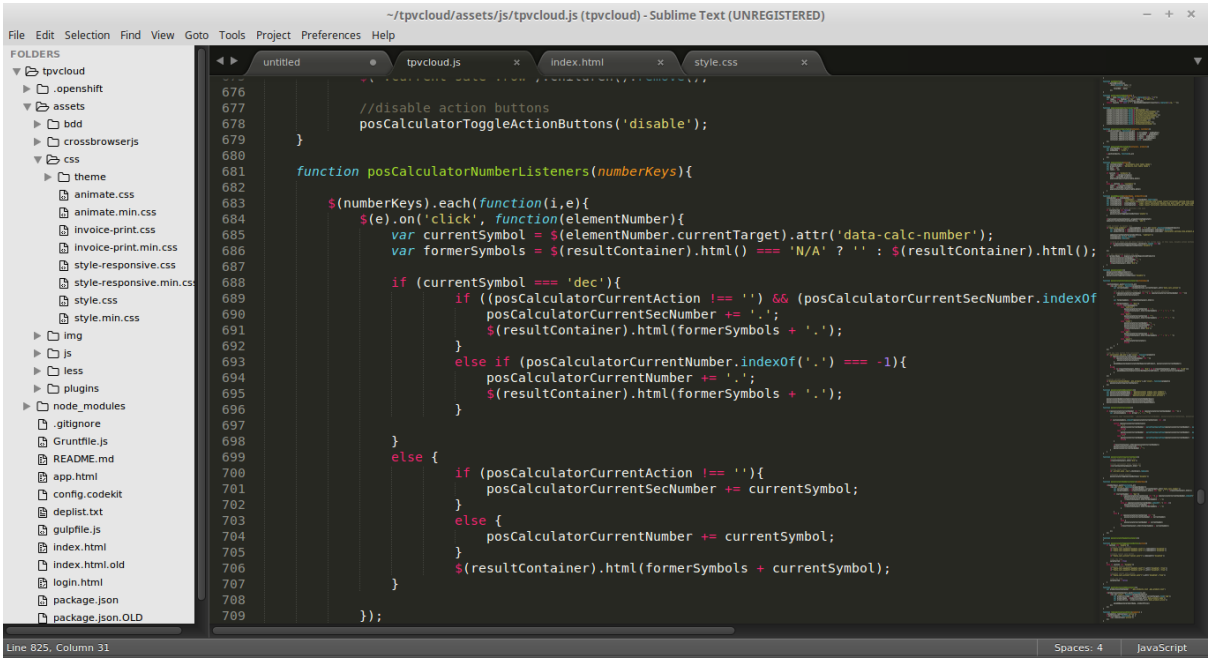
- Consola "DevTools" de "Google Chrome"



- Consola "Firebug" de "Mozilla Firefox"



- Como editor de texto, “Sublime Text 3”[17]



Capítulo 4: Implementación

1. Requisitos e instalación

La parte que se ha desarrollado, solamente el *front-end*, no requiere ninguna instalación compleja. Para probarlo en local, basta con ejecutar el archivo “login.html” que hay en la raíz de la estructura de la carpeta.

Si se quiere instalar en un servidor web normal, basta con copiar toda la estructura de archivos en donde el servidor requiera.

Se ha probado el correcto funcionamiento en los siguientes sistemas operativos y navegadores:

- Windows 8 con últimas versiones de Google Chrome, Mozilla Firefox e Internet Explorer 10 y 11
- Linux Mint con últimas versiones de Google Chrome y Mozilla Firefox
- Android 4.4 y 5.0 (en Tablet Nexus 7) con últimas versiones de Google Chrome y Mozilla Firefox
- iOS 8 (en iPhone 4S) con últimas versiones de Safari y Google Chrome

Capítulo 5: Demostración

1. Instrucciones de uso

Para acceder a la pantalla de acceso al sistema se ha de hacer a través de la siguiente URL:

http://tpvcloud-uoctfm.rhcloud.com/login.html

Se podrán usar los siguientes usuarios y contraseñas:

USUARIO	CONTRASEÑA
employee1	1employee
employee2	2employee
employee3	3employee
employee4	4employee
Administrator	myadmin

El usuario “administrator” mostrará en el menú lateral la sección de “Administración” en el que se pueden visualizar analíticas, contabilidad y empleados.

En cuando al uso de la aplicación se puede observar un vídeo demostrativo en la siguiente URL:

http://youtu.be/5TxTZr7rbfY

2. Prototipos

Los prototipos se crearon con una herramienta Cloud llamada *lucidchart* (lucidchart.com). Debido al volumen grande de las imágenes y que los mismos son interactivos, se indica la URL a la que se puede acceder para ver e interactuar con los mismos: http://cort.as/O6ZZ

Capítulo 6: Conclusiones y líneas de futuro

1. Conclusiones

La principal conclusión es que es complicado plasmar en un producto final lo que se piensa al principio. Es muy complejo llevar siempre a cabo las ideas iniciales.

En el caso de este proyecto, se realizaron unas maquetas (prototipos) de una forma y finalmente, el producto final dista bastante de esas maquetas.

A medida que uno avanza en el desarrollo se encuentra con escollos del desarrollo y con problemas de usabilidad que no se planteaban en un primer momento. Cuando se comienza “a jugar” con la aplicación en el navegador, no siempre las primeras ideas ni los prototipos atinan con un uso fácil y claro.

Por lo tanto, todo esto hace que la planificación siempre se desvíe. Dar una fecha concreta de entrega de una parte, parece algo complejo, un viejo problema de la gestión de proyectos que, como mínimo, siempre tiene un desvío del 20-30% del tiempo planteado inicialmente.

En cuanto a los objetivos, no se han podido conseguir todos y quedan diferentes puntos por acabar e implementar en el futuro, debido a una no muy buena planificación del proyecto y a otros problemas surgidos (en otro ámbito) en el desarrollo del mismo.

La metodología “Kanban” se considera la correcta y no ha sido obstáculo en el no cumplimiento de las fechas.

Por norma general en un principio se ve de una forma y a medida que se avanza se ha de ir dejando por el camino algunos aspectos a los que no se le pueden dedicar tiempo y centrarse en lo importante, en la funcionalidad principal.

2. Líneas de futuro

En cuanto a lo que faltaría por desarrollar sería lo siguiente:

- Gestión de categorías de productos. Podre crear, editar y eliminar. Poder en la pantalla principal filtrar productos por categorías
- Buscador de productos y de clientes
- Integrar con alguna pasarela de pago
- Cambio de contraseña de un empleado
- Poder enlazar una venta con un activo (p.e. con una mesa de un restaurante)
- Aplicar descuento a una venta
- Crear un *workflow* para poder enviar datos de contabilidad por correo electrónico (p.e. a un gestor de la empresa)
- Poder enlazar la TPV con sistemas genéricos de tiendas *online* como “Prestashop” o “Magento”

Bibliografía

[1] *"Terminal punto de Venta"*, http://cort.as/O4Ov, (Septiembre 2014)

[2] *"What is SaaS in Cloud Computing?"* , http://cort.as/O4PE, (Septiembre 2014)

[3] *"Monetize Apps: Paid Apps vs. In-App Purchases vs. Freemium vs. Subscription ",* http://cort.as/O4PQ, (Octubre 2014)

[4] "*SEM: google adwords*", http://cort.as/O4Pg , (Enero 2015)

[5] "*Mejora tu trabajo en equipo con Kanban*", http://cort.as/O4Pp, (Enero 2015)

[6] "*What is front-end development*", http://cort.as/O4QZ, (Octubre 2014)

[7] *Bootstrap 3*, http://cort.as/O4Qi, (Enero 2015)

[8] "*What is a framework? What does it do? Why do we need a framework*", http://cort.as/O4Rq, (Noviembre 2014)

[9] "*What is cloud computing?*", http://cort.as/O4dO, (Octubre 2014)

[10] "*¿Qué son los lenguajes dinámicos?*", http://cort.as/O4e3, (Octubre 2014)

[11] "*The growth of dynamic languages*", http://cort.as/O4ee, (Octubre 2014)

[12] "*Languages comparison per Project*", http://cort.as/8Vqa, (Octubre 2014)

[13] "*Retrato PYME 2014 en el Ministerio de Industria, Energía y Turimo*", http://cort.as/O5kF, (Diciembre 2014)

[14] "*Unidades de medida en diseño web*", http://cort.as/O6lE, (Enero 2015)

[15] jQuery, http://cort.as/O7mG , (Enero 2015)

[16] chart.js, http://cort.as/O7mK , (Enero 2015)

[17] modernizr.js , http://cort.as/O7mL , (Enero 2015)

[18] sublimetext 3 , http://cort.as/O7mP , (Enero 2015)

Anexos

Anexo A: Glosario

App nativa: aplicación desarrollada para que funcione exclusivamente en un sistema operativo móvil (Android, iOS, etc…).

Back-end: En una arquitectura de cliente-servidor, es la parte del servidor.

Bootstrap: Librería para desarrollo web con componentes reusables.

CSS3: Hojas de estilo en cascada versión 3. Usadas para dar estilo (colores, formas, etc..) a las páginas web

DevTools: Herramienta del navegador "Google Chrome" para uso de desarrolladores *web*.

Firebug: Herramienta del navegador "Mozilla Firefox" para uso de desarrolladores *web*.

Freemium: Modelo de negocio que brinda servicios básicos gratuitos y se cobran por otros más avanzados.

Front-end: En una arquitectura de cliente-servidor, es la parte del cliente.

Framework: Bibliotecas que brindan facilidades de uso a los lenguajes de programación.

HTML5: Lenguaje de etiquetas para realizar páginas *web*.

IaaS: *Infraestructure as a Service*, modelo de distribución de infraestructura tecnológica. Normalmente son máquinas virtuales en otros servidores, que se pueden configurar a antojo del usuario. Se suele tener total control como administrador.

javaScript: lenguaje de programación interpretado convertido en el lenguaje casi "de facto" en el lado cliente (en programación *web*).

JSON: *javaScript Object Notation*, modelo de intercambio de datos de forma ligera que es un subconjunto de *javaScript*.

Kanban: Del japonés "tarjeta visual" , es un sistema de organización de la producción mediante un sistema de tarjetas con tareas que van cambiando en el tiempo de proceso.

Mapa de navegación: Representación gráfica de la organización de una *web*.

MBaaS: *Mobile Back-end as a Service*, modelo alojado en la "nube" para proporcionar a desarrolladores *web* y de aplicaciones móviles una forma de vincular diferentes servicios como analíticas, notificaciones, etc…

Mind map: Diagrama para representar modelos o conceptos alrededor de una idea central.

Nube/Cloud: Paradigma que ofrece servicios a través de Internet.

Openshift: PaaS de Red Hat con soporte para multitud de lenguajes de programación y bases de datos.

PaaS: *Platform as a Service*, plataforma que ofrece todo tipo de servicios sin tener que configurar aspectos de sistemas operativos (como se haría en un IaaS). Suelen proveer de controles sencillos para deplegar aplicaciones *web*.

PYME/SME: Pequeña y mediana empresa (*small and medium-sized enterprises* en inglés), empresas hasta 249 empleados (en España).

Responsive: Diseño de páginas web que se adapta a dispositivos sean cual sea el tamaño de la pantalla.

SaaS: *Software as a Service*, modelo de distribución de *software* alojado en otros servidores que no son los propios, normalmente en la "nube".

SEM: *Search engine marketing*, método para promover páginas *web* en los buscadores mediante el pago por colocación.

SEO: *Search engine optimization*, método para posicionar páginas *web* en buscadores mediante técnicas de desarrollo.

Sublime Text 3: Potente editor de texto para desarrollo con multitud de *plug-ins*. Ligero, flexible y multiplataforma.

TPV/POS: Terminal punto de venta (*Point of sale* en inglés) puede referirse a un *hardware* o *software* dedicado al registro de ventas en un establecimiento.

Web hosting: Servicio para poder alojar páginas web sin apenas control sobre algunas características. Normalmente está cerrado a desarrollo con “HTML5-CSS3-javaScript” y “PHP”.

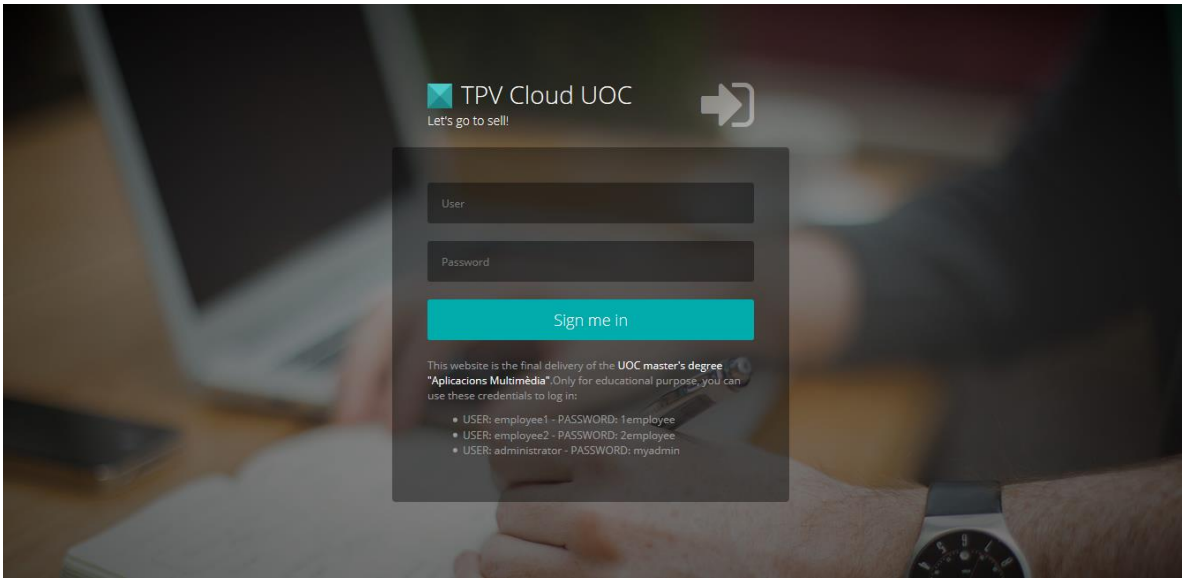
Anexo B: Entregables

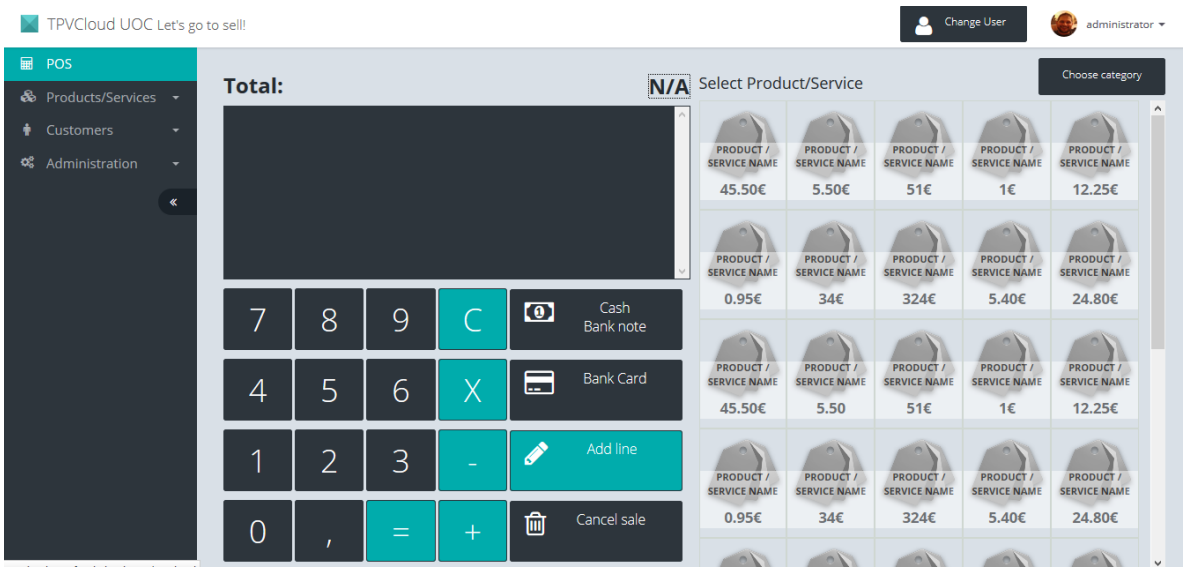
Con la memoria se entregan los siguientes documentos:

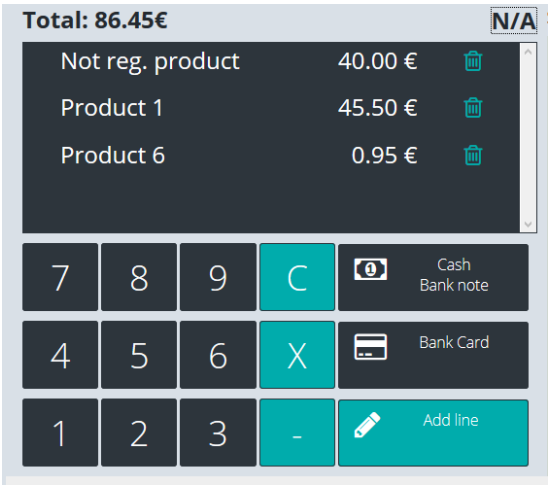
- cgarde_presentacion.pdf : Archivo con la presentación del proyecto. Incluye enlace a vídeo de demostración de uso
- Tpvcloud: carpeta con todo el código “HTML”, “CSS” y “javaScript” del proyecto. Simplemente se ha de ejecutar el archivo login.html

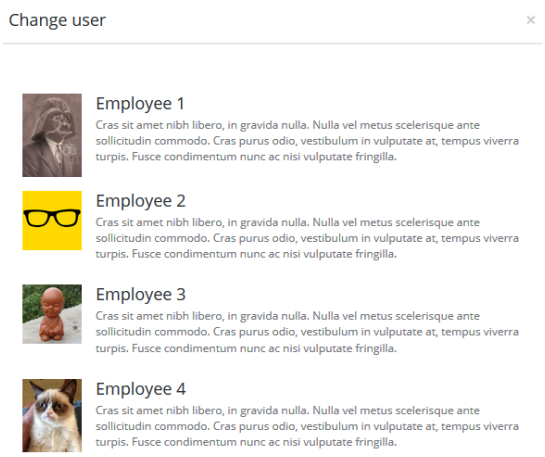
Anexo C: Capturas de pantalla

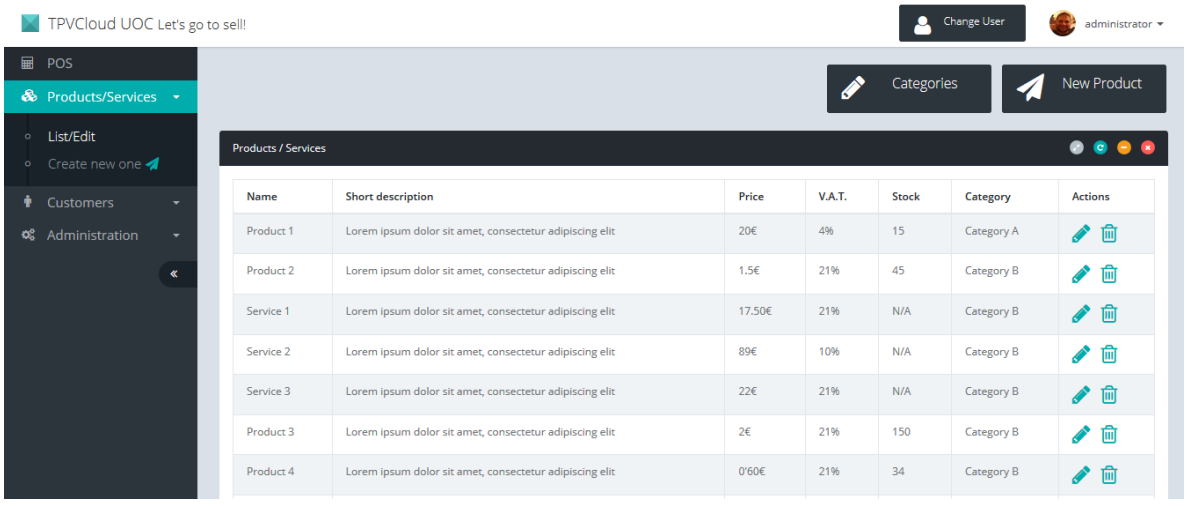
- Resolución de portátil estándar (1366x768px):

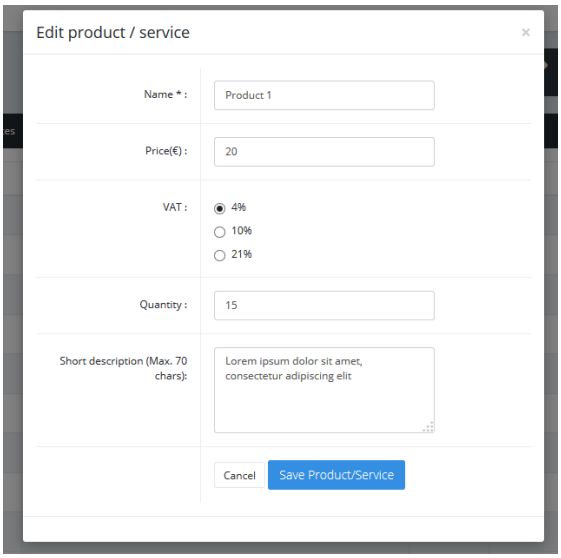


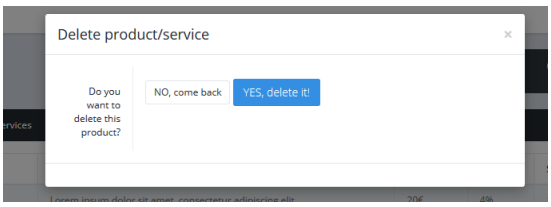

TPVCloud UOC Let's go to sell!
Change User
administrator
POS
Products/Services
Customers
Administration
Total:
N/A
Select Product/Service
Choose category
PRODUCT / SERVICE NAME
45.50€
5.50€
51€
1€
12.25€
0.95€
34€
324€
5.40€
24.80€
5.50
7 8 9 C
4 5 6 X
1 2 3 -
0 , = +
Cash Bank note
Bank Card
Add line
Cancel sale

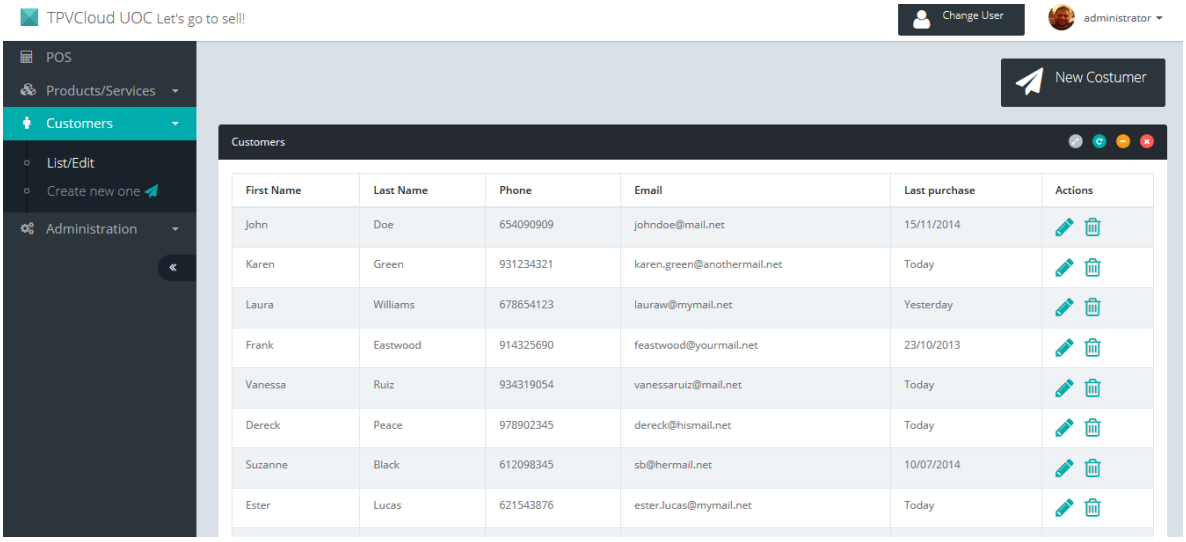

Total: 86.45€
N/A
Not reg. product 40.00 €
Product 1 45.50 €
Product 6 0.95 €
7 8 9 C
4 5 6 X
1 2 3 -
Cash Bank note
Bank Card
Add line

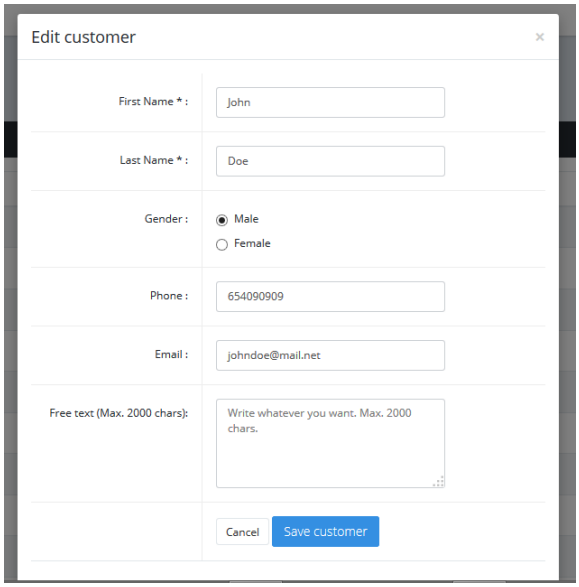

Change user
Employee 1
Cras sit amet nibh libero, in gravida nulla. Nulla vel metus scelerisque ante sollicitudin commodo. Cras purus odio, vestibulum in vulputate at, tempus viverra turpis. Fusce condimentum nunc ac nisi vulputate fringilla.
Employee 2
Cras sit amet nibh libero, in gravida nulla. Nulla vel metus scelerisque ante sollicitudin commodo. Cras purus odio, vestibulum in vulputate at, tempus viverra turpis. Fusce condimentum nunc ac nisi vulputate fringilla.
Employee 3
Cras sit amet nibh libero, in gravida nulla. Nulla vel metus scelerisque ante sollicitudin commodo. Cras purus odio, vestibulum in vulputate at, tempus viverra turpis. Fusce condimentum nunc ac nisi vulputate fringilla.
Employee 4
Cras sit amet nibh libero, in gravida nulla. Nulla vel metus scelerisque ante sollicitudin commodo. Cras purus odio, vestibulum in vulputate at, tempus viverra turpis. Fusce condimentum nunc ac nisi vulputate fringilla.


TPVCloud UOC Let's go to sell!
Change User
administrator
POS
Products/Services
List/Edit
Create new one
Customers
Administration
Categories
New Product
Products / Services
Name
Short description
Price
V.A.T.
Stock
Category
Actions
Product 1
Lorem ipsum dolor sit amet, consectetur adipiscing elit
20€
4%
15
Category A
Product 2
1.5€
21%
45
Category B
Service 1
17.50€
N/A
Service 2
89€
10%
Service 3
22€
Product 3
2€
150
Product 4
0'60€
34

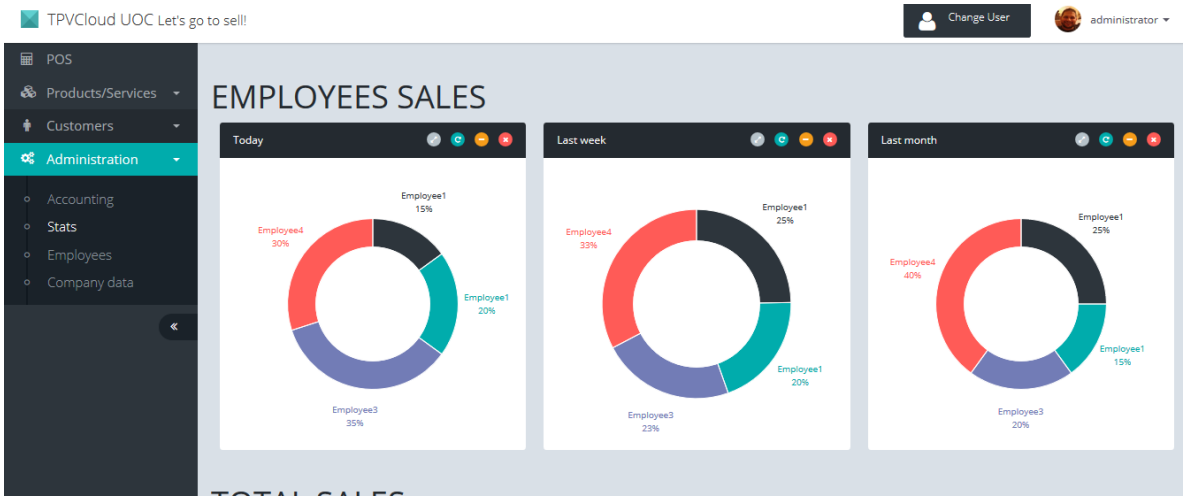

Edit product / service
Name * :
Product 1
Price(€) :
20
VAT :
4%
10%
21%
Quantity :
15
Short description (Max. 70 chars):
Lorem ipsum dolor sit amet, consectetur adipiscing elit
Cancel
Save Product/Service

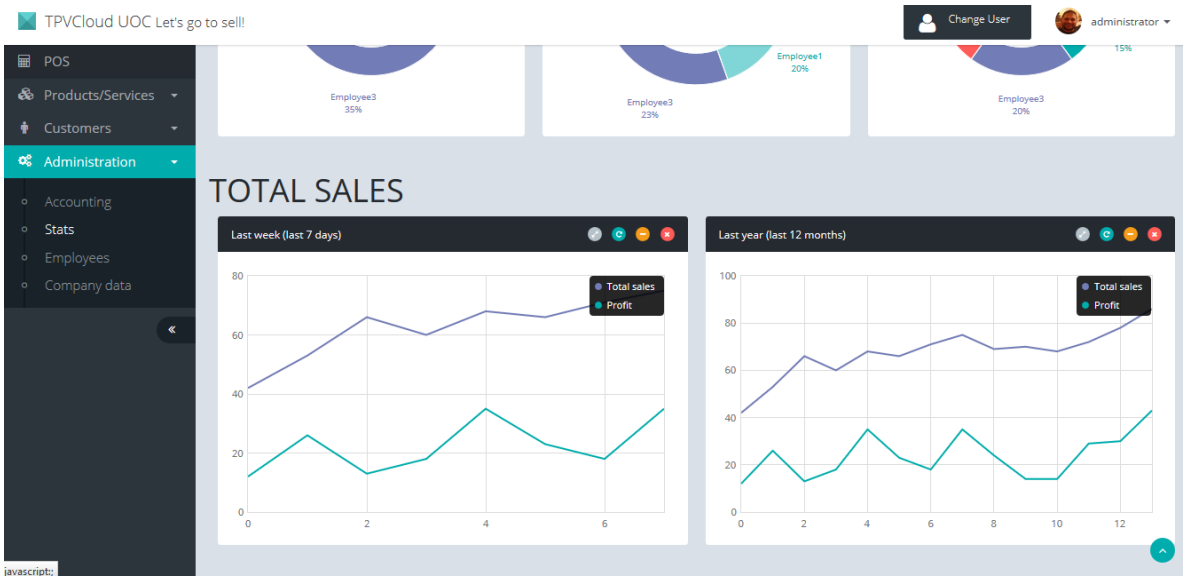

Delete product/service
Do you want to delete this product?
NO, come back
YES, delete it!

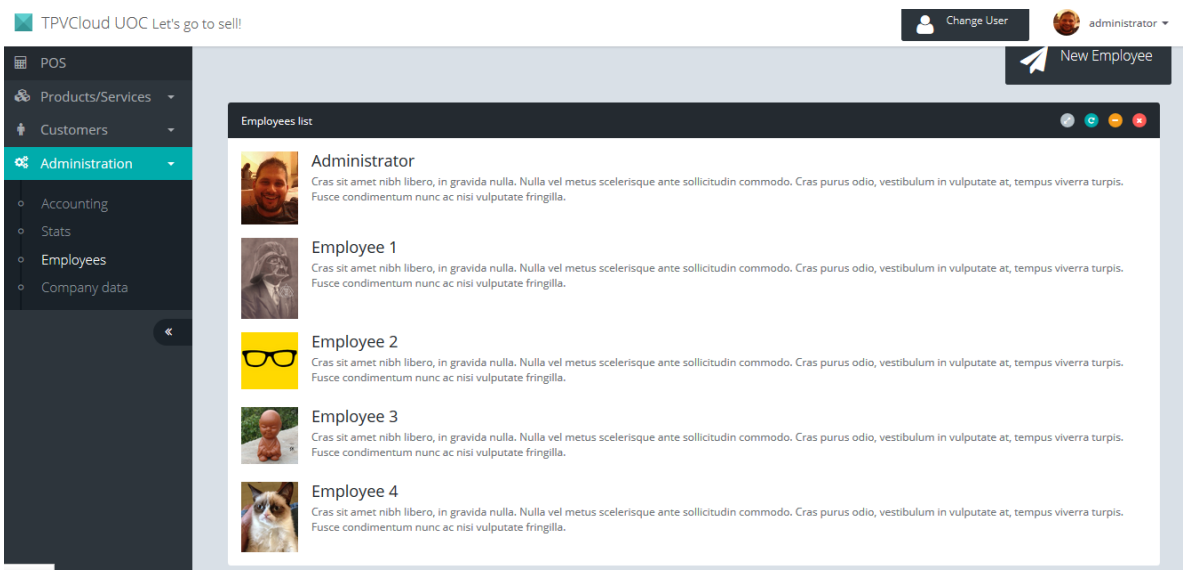

TPVCloud UOC Let's go to sell!
Change User
administrator
POS
Products/Services
Customers
List/Edit
Create new one
Administration
New Costumer
Customers
First Name
Last Name
Phone
Email
Last purchase
Actions
John Doe 654090909 johndoe@mail.net 15/11/2014
Karen Green 931234321 karen.green@anothermail.net Today
Laura Williams 678654123 lauraw@mymail.net Yesterday
Frank Eastwood 914325690 feastwood@yourmail.net 23/10/2013
Vanessa Ruiz 934319054 vanessaruiz@mail.net Today
Dereck Peace 978902345 dereck@hismail.net Today
Suzanne Black 612098345 sb@hermail.net 10/07/2014
Ester Lucas 621543876 ester.lucas@mymail.net Today


Edit customer
First Name * :
John
Last Name * :
Doe
Gender :
Male
Female
Phone :
654090909
Email :
johndoe@mail.net
Free text (Max. 2000 chars):
Write whatever you want. Max. 2000 chars.
Cancel
Save customer

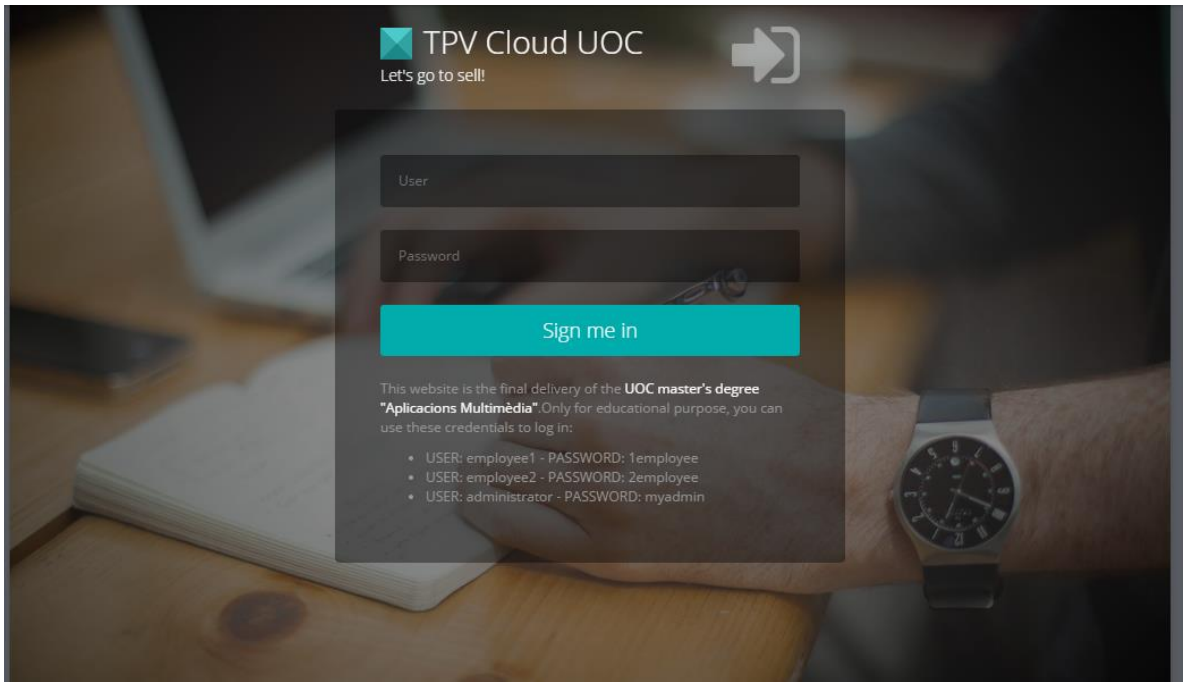

POS
Products/Services
Customers
Administration
Accounting
Stats
Employees
Company data
TODAY
SELLS
8.900€
Better than last yesterday (20.1%)
PROFIT
1.200€
Better than yesterday (10.5%)
OUTGOINGS
900€
More than yesterday (6.3%)
PRODUCTS/SERVICES SOLD
123
Better than yesterday (4.9%)
LAST WEEK
SELLS
28.900€
Better than last week (20.1%)
PROFIT
7.200€
Better than last week (10.5%)
OUTGOINGS
4.200€
More than last week (6.3%)
PRODUCTS/SERVICES SOLD
746
Better than last week (4.9%)
LAST MONTH

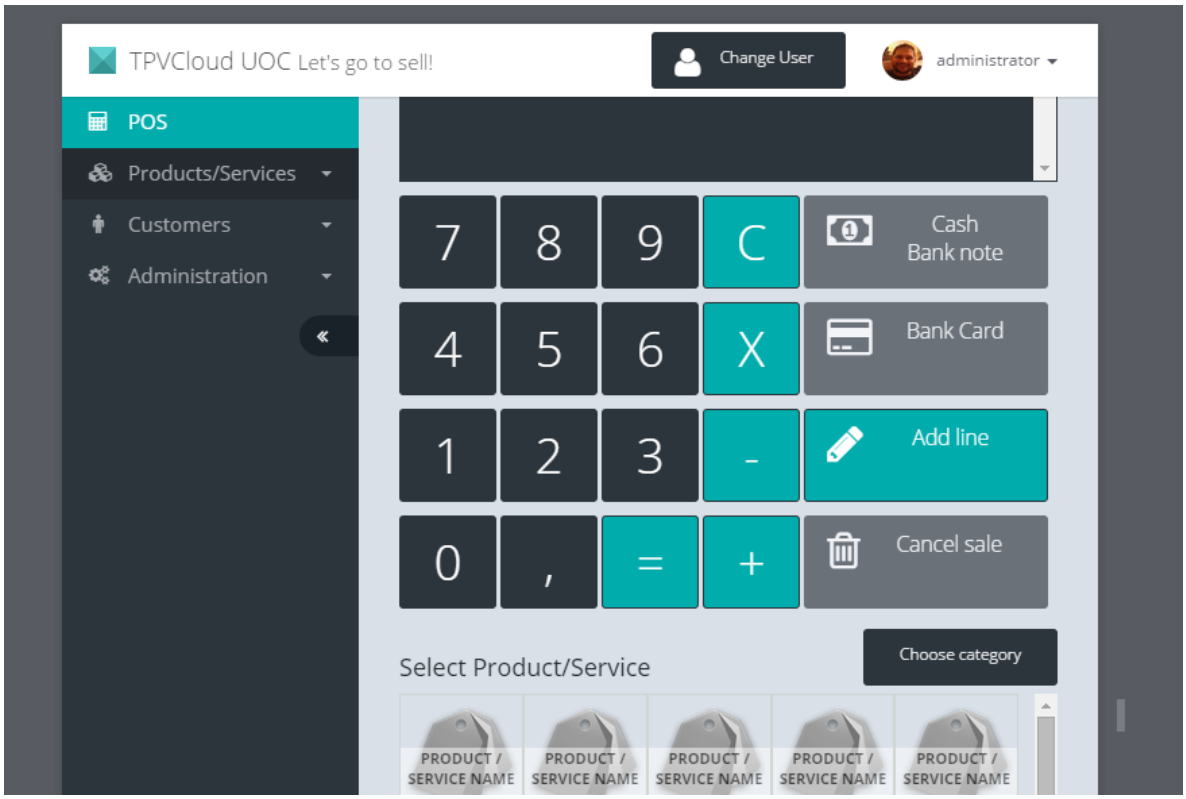

TPVCloud UOC Let's go to sell!
Change User
administrator
POS
Products/Services
Customers
Administration
Accounting
Stats
Employees
Company data
EMPLOYEES SALES
Today
Employee1 15%
Employee4 30%
Employee1 20%
Employee3 35%
Last week
Employee1 25%
Employee4 33%
Employee1 20%
Employee3 23%
Last month
Employee1 25%
Employee4 40%
Employee1 15%
Employee3 20%

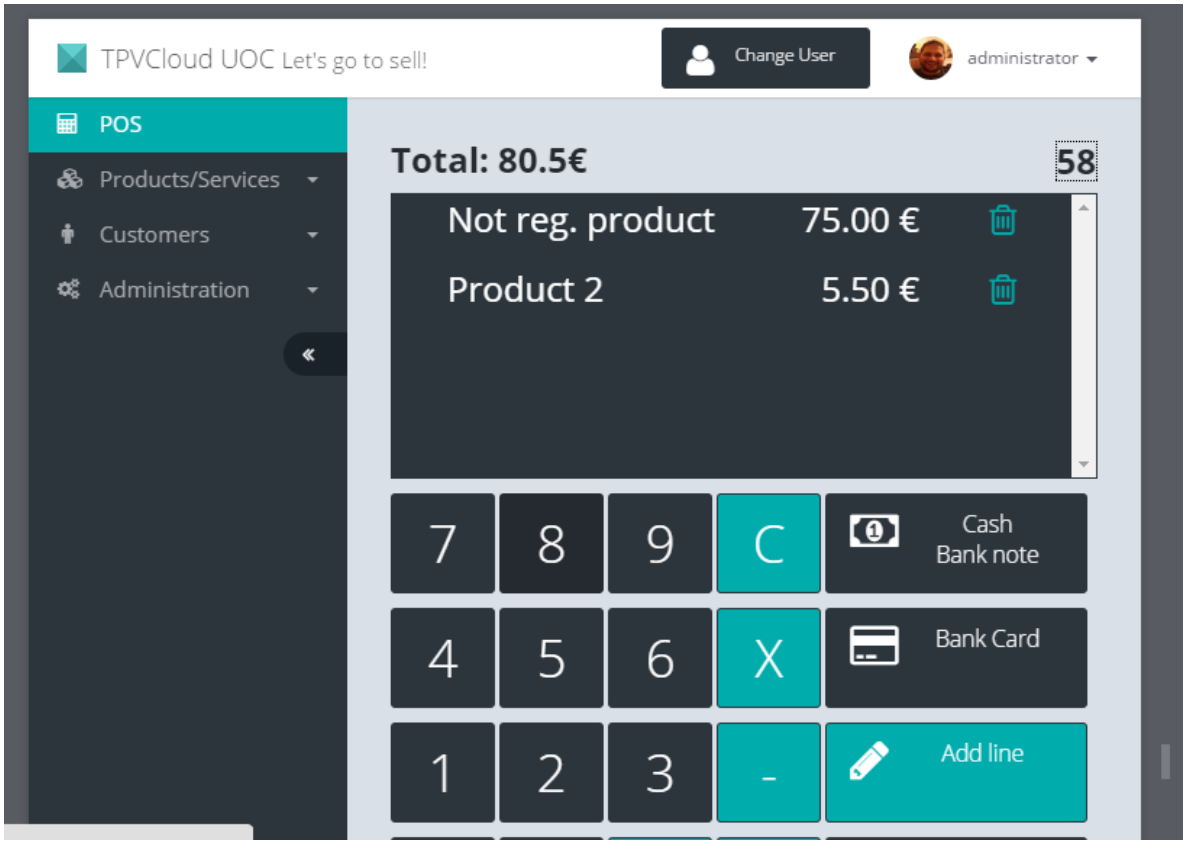

TPVCloud UOC Let's go to sell!
Change User
administrator
POS
Products/Services
Customers
Administration
Accounting
Stats
Employees
Company data
TOTAL SALES
Last week (last 7 days)
Total sales
Profit
Last year (last 12 months)
Total sales
Profit
javascript:;

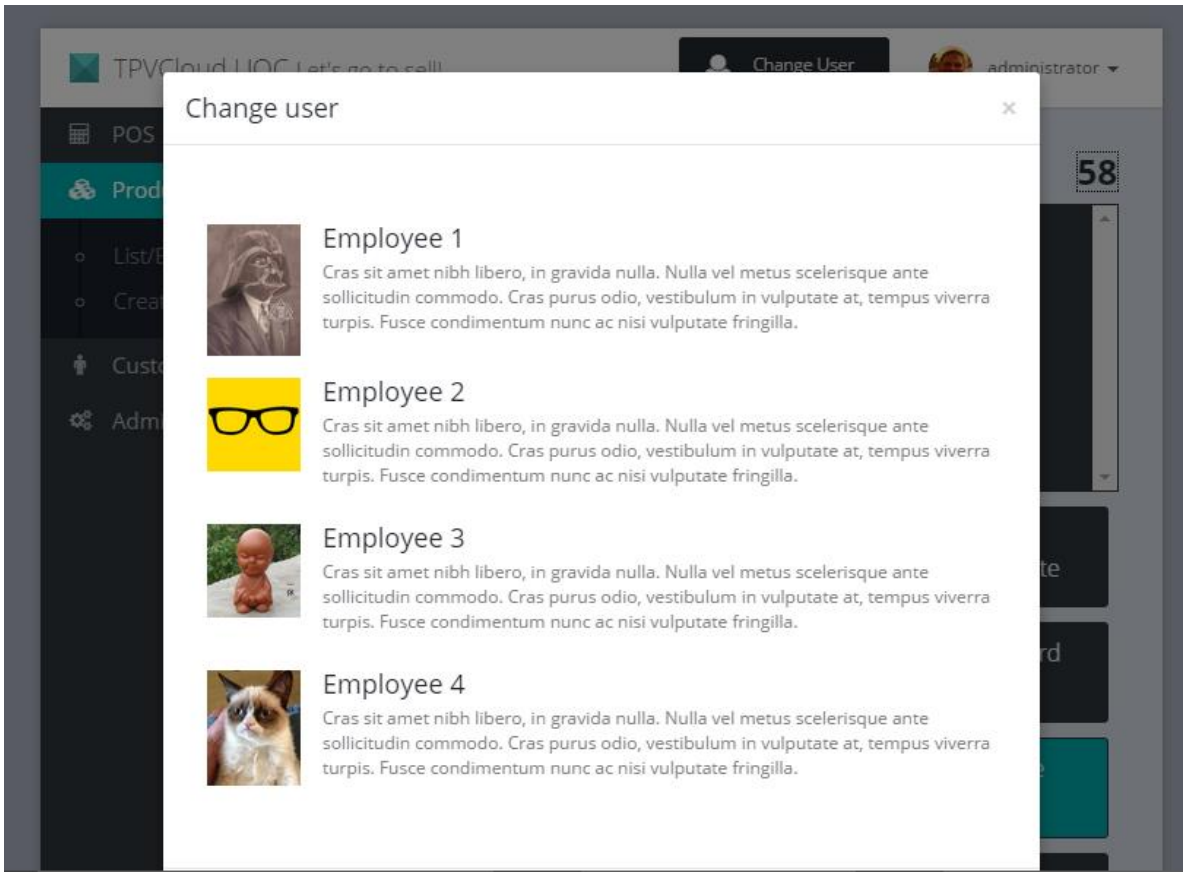


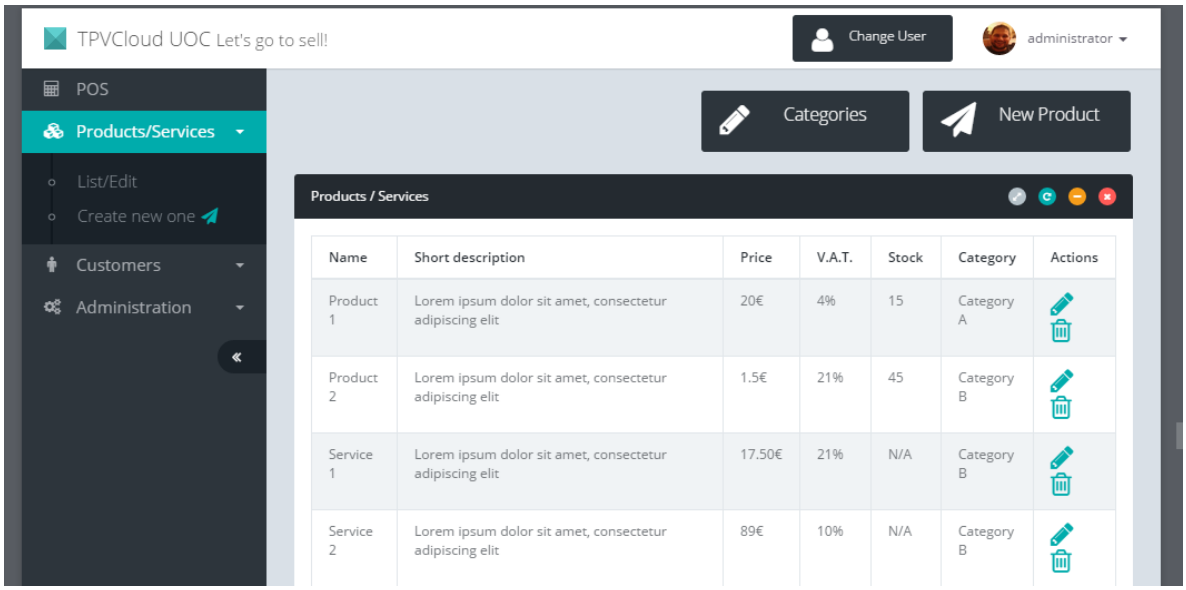
- Resolución Tablet estándar (1024x768px):

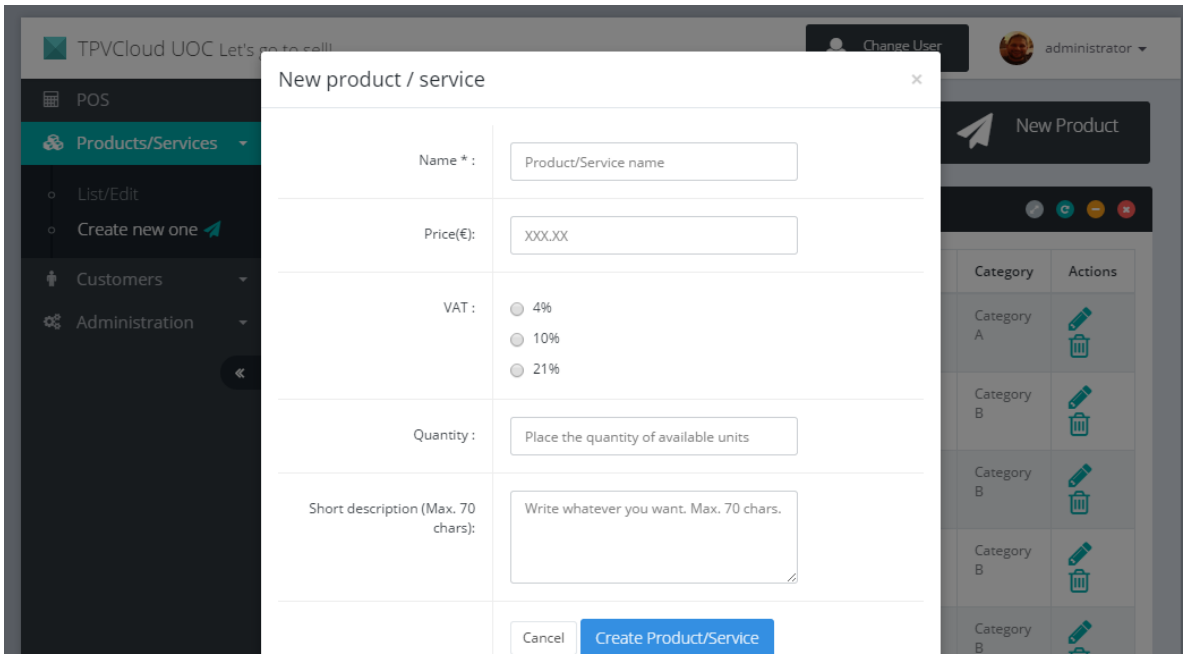


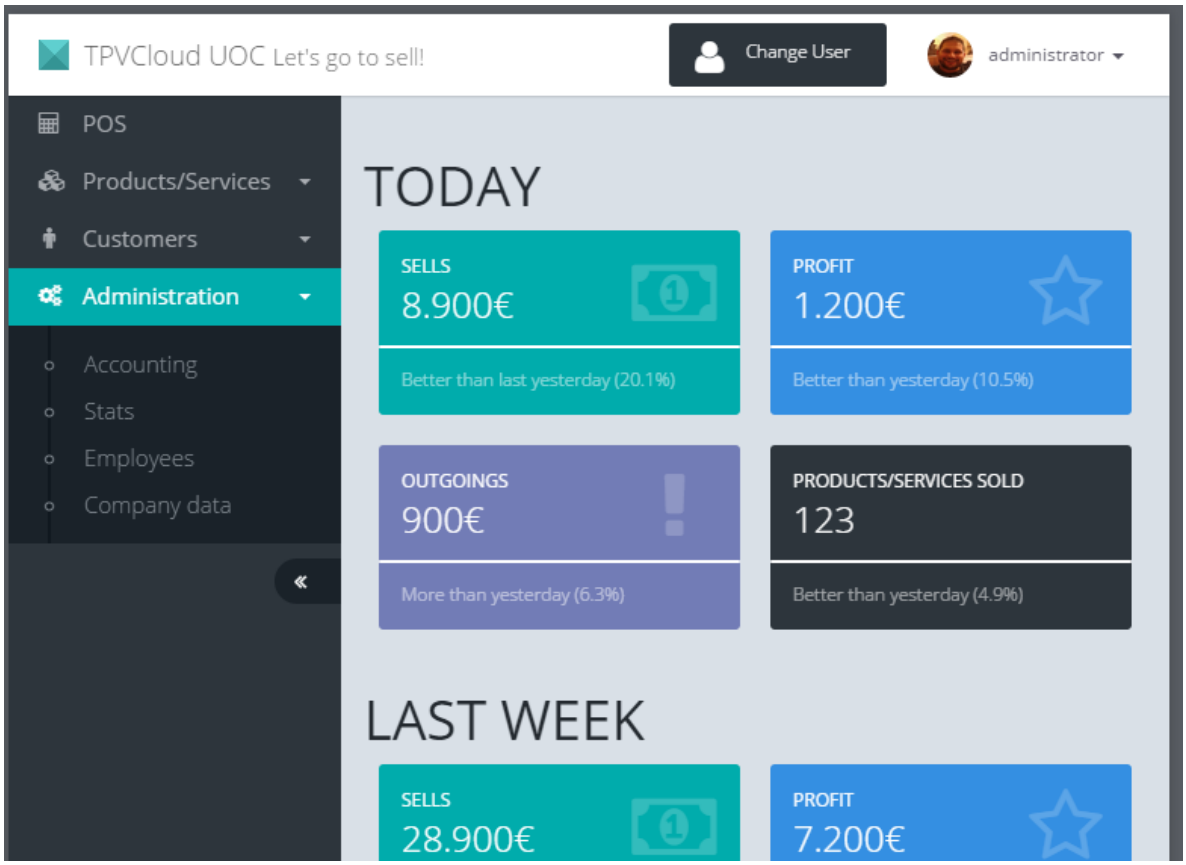

TPVCloud UOC Let's go to sell!
Change User
administrator
POS
Products/Services
Customers
Administration
7
8
9
C
Cash
Bank note
4
5
6
X
Bank Card
1
2
3
-
Add line
0
,
=
+
Cancel sale
Select Product/Service
Choose category
PRODUCT / SERVICE NAME
PRODUCT / SERVICE NAME
PRODUCT / SERVICE NAME
PRODUCT / SERVICE NAME
PRODUCT / SERVICE NAME

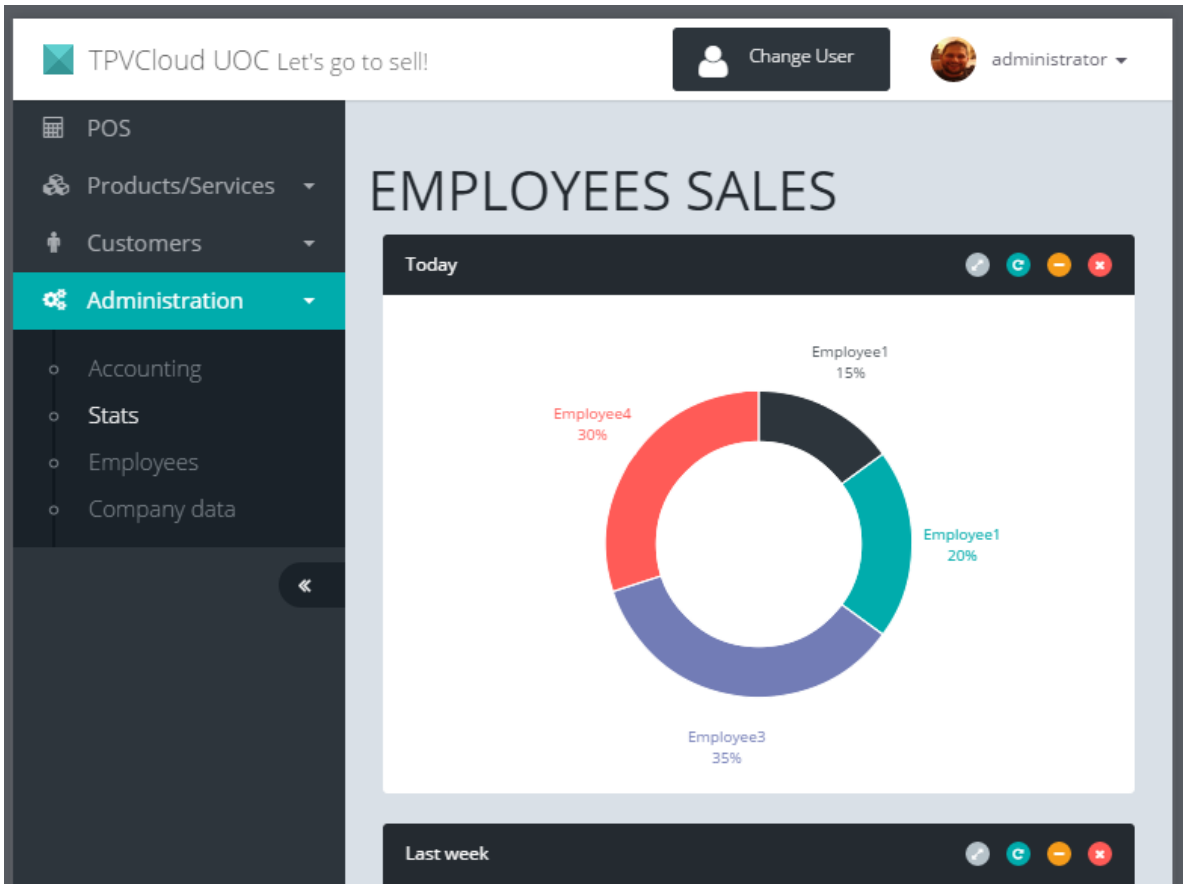

TPVCloud UOC Let's go to sell!
Change User
administrator
POS
Products/Services
Customers
Administration
Total: 80.5€
58
Not reg. product
75.00 €
Product 2
5.50 €
7
8
9
C
Cash
Bank note
4
5
6
X
Bank Card
1
2
3
-
Add line

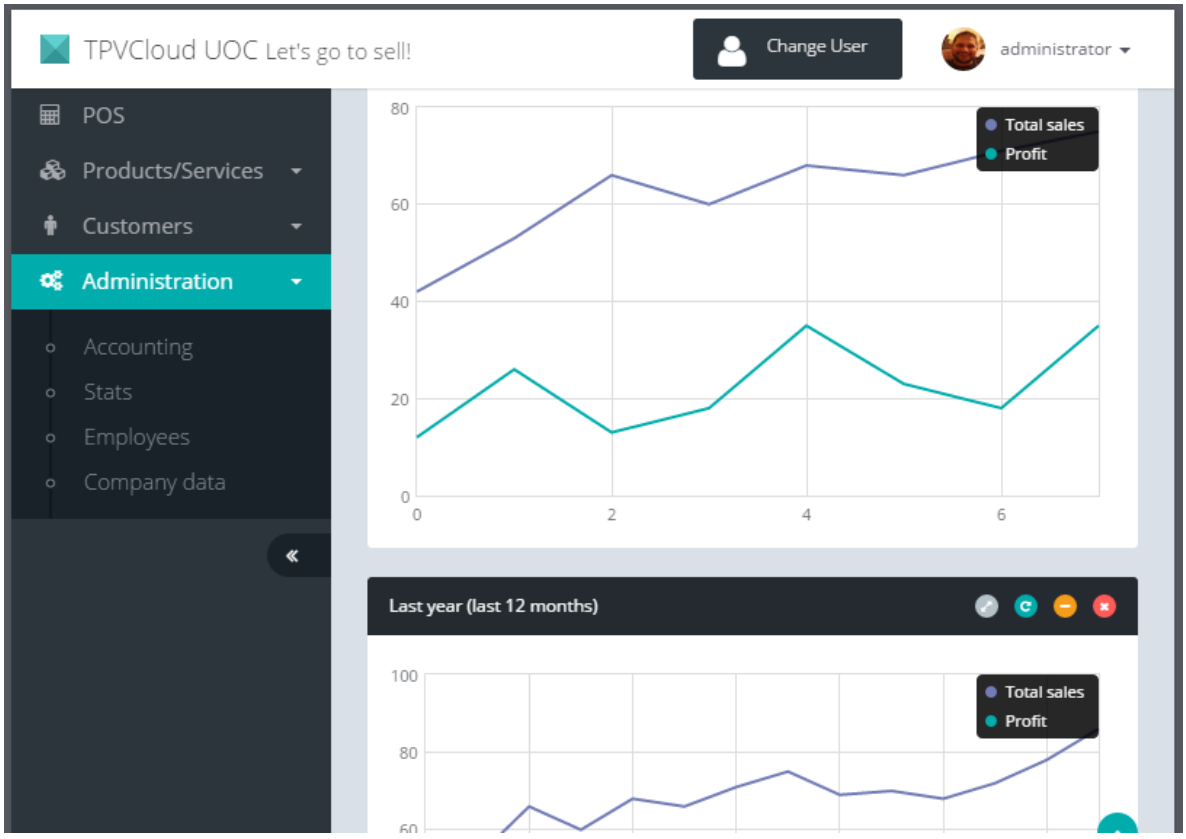

Change user
Employee 1
Cras sit amet nibh libero, in gravida nulla. Nulla vel metus scelerisque ante sollicitudin commodo. Cras purus odio, vestibulum in vulputate at, tempus viverra turpis. Fusce condimentum nunc ac nisi vulputate fringilla.
Employee 2
Cras sit amet nibh libero, in gravida nulla. Nulla vel metus scelerisque ante sollicitudin commodo. Cras purus odio, vestibulum in vulputate at, tempus viverra turpis. Fusce condimentum nunc ac nisi vulputate fringilla.
Employee 3
Cras sit amet nibh libero, in gravida nulla. Nulla vel metus scelerisque ante sollicitudin commodo. Cras purus odio, vestibulum in vulputate at, tempus viverra turpis. Fusce condimentum nunc ac nisi vulputate fringilla.
Employee 4
Cras sit amet nibh libero, in gravida nulla. Nulla vel metus scelerisque ante sollicitudin commodo. Cras purus odio, vestibulum in vulputate at, tempus viverra turpis. Fusce condimentum nunc ac nisi vulputate fringilla.

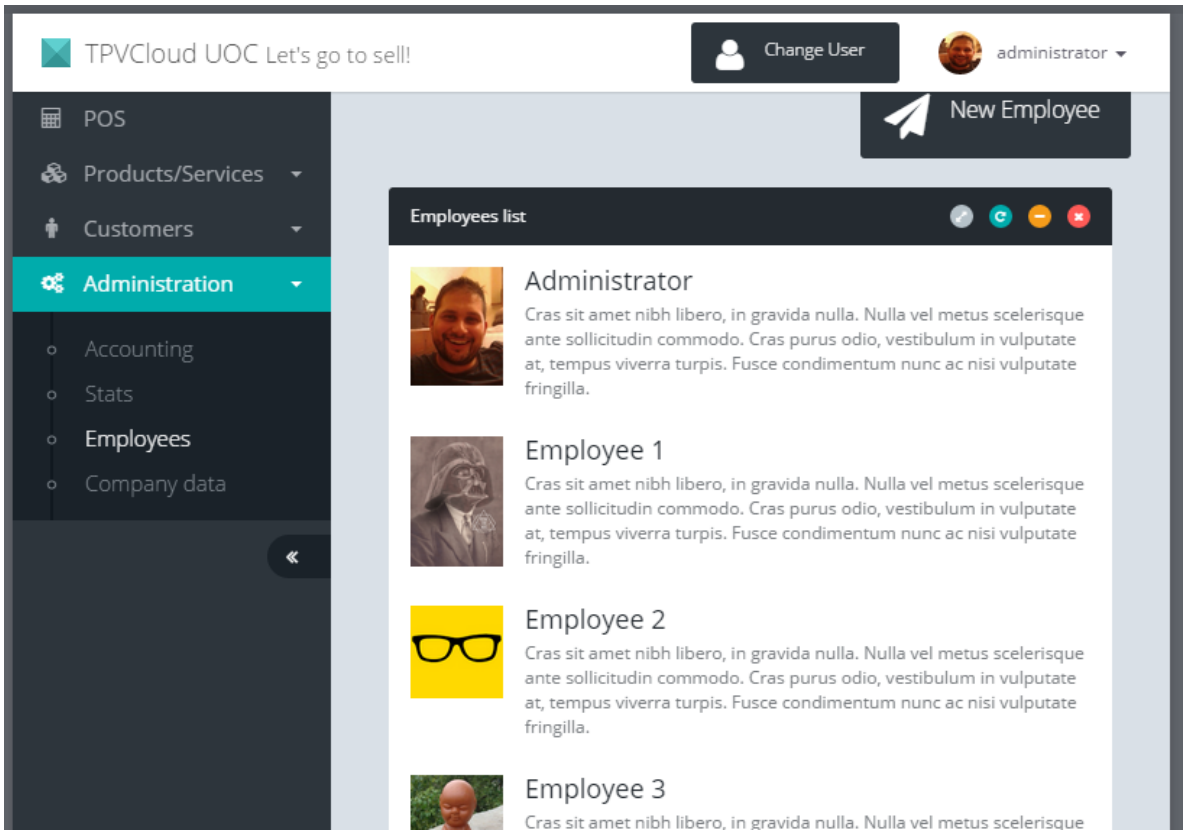

TPVCloud UOC Let's go to sell!
Change User
administrator
POS
Products/Services
List/Edit
Create new one
Customers
Administration
Categories
New Product
Products / Services
Name
Short description
Price
V.A.T.
Stock
Category
Actions
Product 1
Lorem ipsum dolor sit amet, consectetur adipiscing elit
20€
4%
15
Category A
Product 2
Lorem ipsum dolor sit amet, consectetur adipiscing elit
1.5€
21%
45
Category B
Service 1
Lorem ipsum dolor sit amet, consectetur adipiscing elit
17.50€
21%
N/A
Category B
Service 2
Lorem ipsum dolor sit amet, consectetur adipiscing elit
89€
10%
N/A
Category B


TPVCloud UOC Let's go to sell!
Change User
administrator
POS
Products/Services
List/Edit
Create new one
Customers
Administration
New Product
Category
Actions
Category A
Category B
Category B
Category B
Category B
New product / service
Name * :
Product/Service name
Price(€):
XXX.XX
VAT :
4%
10%
21%
Quantity :
Place the quantity of available units
Short description (Max. 70 chars):
Write whatever you want. Max. 70 chars.
Cancel
Create Product/Service

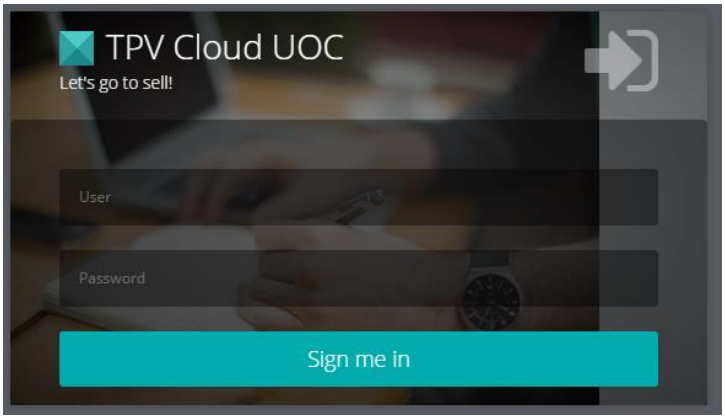

TPVCloud UOC Let's go to sell!
Change User
administrator
POS
Products/Services
Customers
Administration
Accounting
Stats
Employees
Company data
TODAY
SELLS
8.900€
Better than last yesterday (20.1%)
PROFIT
1.200€
Better than yesterday (10.5%)
OUTGOINGS
900€
More than yesterday (6.3%)
PRODUCTS/SERVICES SOLD
123
Better than yesterday (4.9%)
LAST WEEK
SELLS
28.900€
PROFIT
7.200€

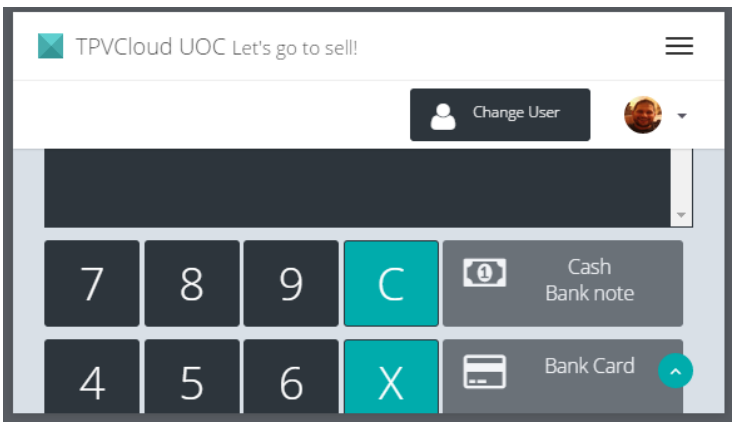

TPVCloud UOC Let's go to sell!
Change User
administrator
POS
Products/Services
Customers
Administration
Accounting
Stats
Employees
Company data
EMPLOYEES SALES
Today
Employee1
15%
Employee4
30%
Employee1
20%
Employee3
35%
Last week

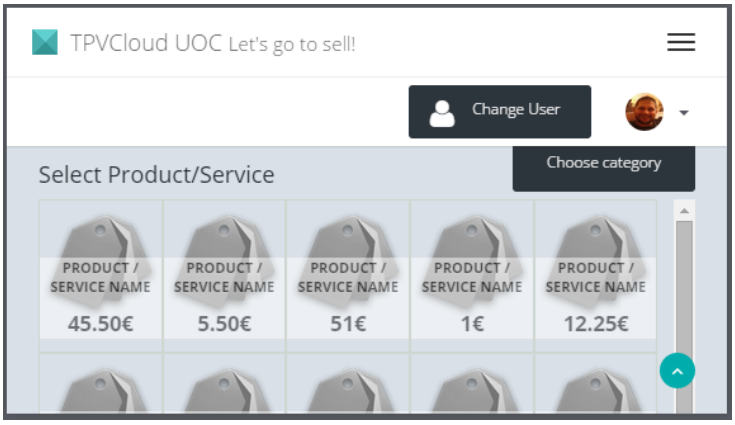

TPVCloud UOC Let's go to sell!
Change User
administrator
POS
Products/Services
Customers
Administration
Accounting
Stats
Employees
Company data
80
60
40
20
0
0
2
4
6
Total sales
Profit
Last year (last 12 months)
100
80
60
Total sales
Profit

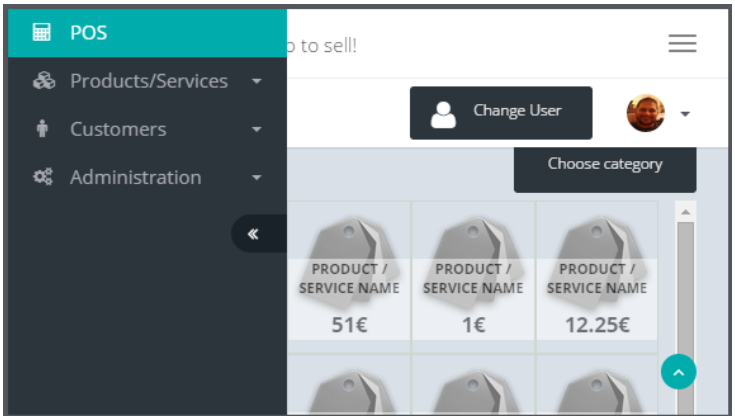


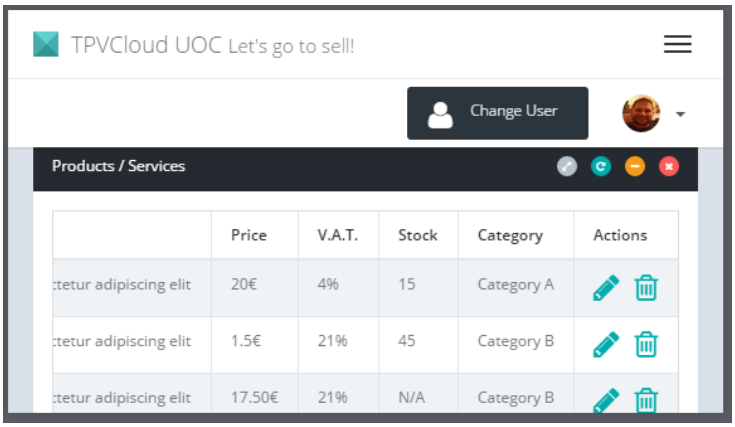
- Resolución móvil pantalla 5 pulgadas (565x320px):

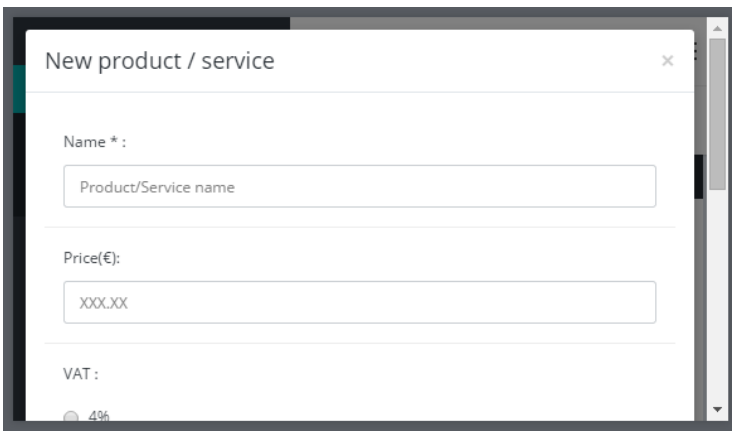


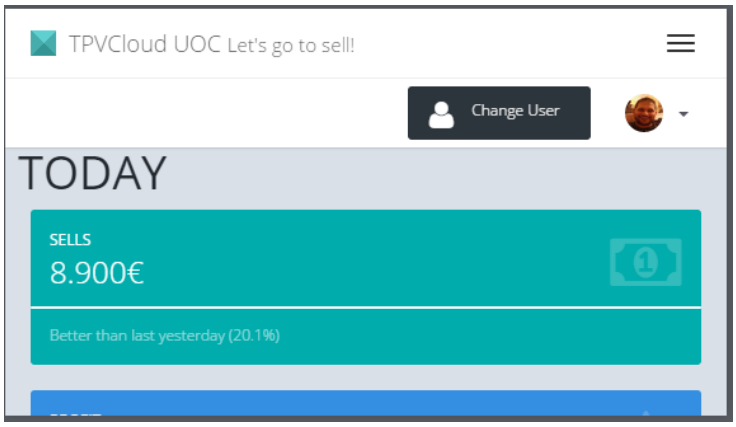


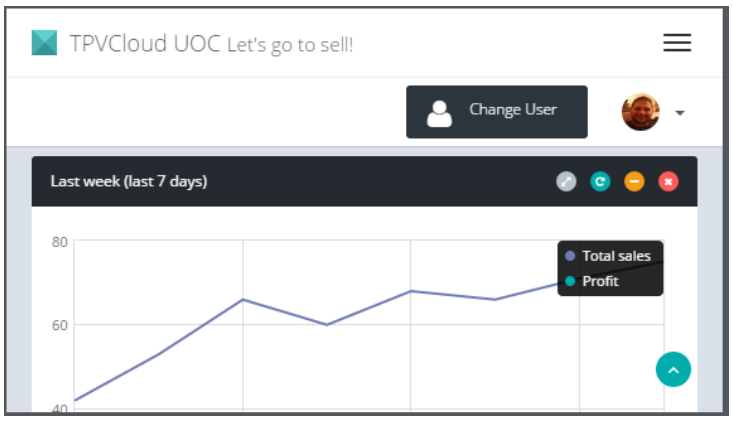

TPVCloud UOC Let's go to sell!
Change User
Select Product/Service
Choose category
PRODUCT / SERVICE NAME
45.50€
5.50€
51€
1€
12.25€

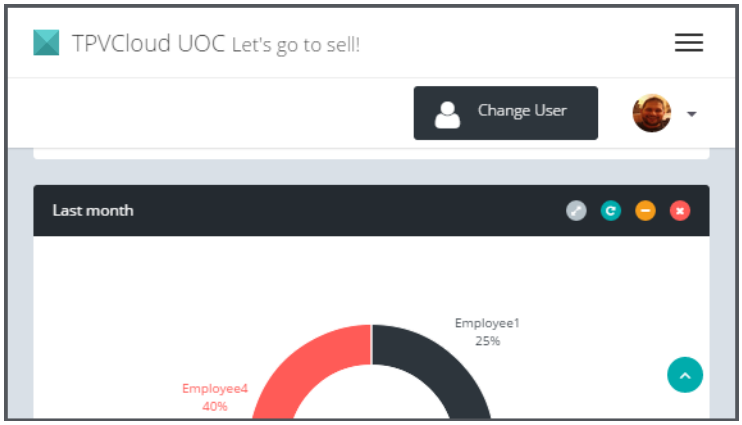

POS
Products/Services
Customers
Administration
Change User
Choose category
PRODUCT / SERVICE NAME
51€
1€
12.25€


TPVCloud UOC Let's go to sell!
Change User
Products / Services
Price
V.A.T.
Stock
Category
Actions
ctetur adipiscing elit
20€
4%
15
Category A
ctetur adipiscing elit
1.5€
21%
45
Category B
ctetur adipiscing elit
17.50€
21%
N/A
Category B


New product / service
Name * :
Product/Service name
Price(€):
XXX.XX
VAT :
4%


TPVCloud UOC Let's go to sell!
Change User
TODAY
SELLS
8.900€
Better than last yesterday (20.1%)


TPVCloud UOC Let's go to sell!
Change User
Last week (last 7 days)
80
60
40
Total sales
Profit


TPVCloud UOC Let's go to sell!
Change User
Last month
Employee1
25%
Employee4
40%

Anexo D: Currículum Vitae

Carlos-Javier Garde Marí

12-Julio-1977 – Barcelona.

Ing. Técnico en informática de gestión por la UOC.

He trabajado diez años en el Grupo Z como técnico de sistemas y desarrollador en .NET

Trabajé en una red social de fútbol como desarrollador en .NET y Silverlight.

Creé mi propia empresa junto a 3 socios de desarrollo web, ampliando mis conocimientos en desarrollo web, javaScript y Python. Gestioné equipos y me dediqué a la formación de desarrollo web. Después de dos años, cambié mi rumbo vendiendo mi parte de la empresa.

He trabajado como *front-end developer* en la empresa Force Manager y actualmente he empezado a trabajar como *full-stack developer* en una empresa de seguridad informática (SASI.COM.MX).

Amplia curiosidad por el desarrollo, asistiendo a multitud de charlas y conferencias (BcnDevCon, Spain.js, PyConES, grupos meetup en Barcelona de javaScript).