Facultade de Informática
UNIVERSIDADE DA CORUÑA

TRABALLO FIN DE GRAO
GRAO EN ENXEÑARÍA INFORMÁTICA
MENCIÓN EN ENXEÑARÍA DO SOFTWARE

Trocos: unha plataforma para potenciar o comercio local

Estudante: Lois López Valdés
Dirección: Fernando Bellas Permuy
Hugo Díaz Devesa

A Coruña, febreiro de 2021.

Á familia, porque sen eles non sería nada

Agradecementos

En primeiro lugar quero darlle as gracias a miña familia, por ser ese apoio co que sempre puiden contar. En especial a meu pai, do que nunca deixo de aprender e que me mostra o camiño que todos deberíamos seguir. Tamén a mamá, por ensinarme o que é o esforzo e a constancia, oxalá chegue a ter algún día a metade da túa forza.

Non podo olvidarme de meu tío e meu primo Fran, que foron os que me amosaron o mundo da informática e aos que sempre lle estarei eternamente agradecido por iso. Tampouco da miña tía Ana, porque sen ela aínda estaría intentando aprobar Cálculo.

A María, por ser a mellor parella que podía imaxinar. Gracias por acompañarme e aguantarme todo este tempo, e espero seguir crecendo e aprendendo ao teu lado.

A tódolos meus amigos, por estar aí sempre, aínda que a distancia e as circunstancias nos separen. Xa celebraremos todo o que non puidemos celebrar este tempo.

A Hugo, por brindarme esta oportunidade e aconsellarme; e a Fernando, polas palabras de ánimo que tanto necesitei escoitar nalgúns momentos, e polos consellos e confianza depositda en min.

Resumo

O obxectivo deste proxecto consiste no deseño e implementación dunha plataforma que proporcione unha ventá de cara ao mundo do *ecommerce* ao comercio local pequeno e mediano que non se pode permitir o desenvolvemento dunha plataforma propia. Esta plataforma debe permitir a búsqueda e seguimento de comercios por parte dos clientes, así como a consumición das ofertas que estes den de alta; e a emisión e consumición de ofertas, así como o envío de notificacións por parte dos comercios.

O *backend* da plataforma consiste nunha API REST, que se encarga de realizar a lóxica de negocio necesaria. Este servizo impleméntase utilizando *.NET Core*. Para levar a cabo a persitencia de datos utilízase *SQL Server*, unha base de datos relacional. Tanto a API REST coma a base de datos atópanse desplegadas nos servizos de *Azure*.

En canto ao *frontend*, atópase separado en dúas aplicacións móbiles, *Trocos* e *TrocosComercios*, desenvolvidas na plataforma Xamarin empregando a linguaxe de etiquetas *XAML* e C#.

A aplicación *Trocos* será a empregada polos clientes, onde poderán buscar e seguir os distintos comercios que se atopen dados de alta na plataforma, e obter os códigos QR necesarios para consumir as ofertas que estes publiciten.

Por outra parte, a aplicación *TrocosComercios* estará dispoñible para os comercios, onde poderán consultar estadísticas sobre os seus comercios, dar de alta novas ofertas, e escanear os códigos QR dos clientes para consumir ditas ofertas.

The objective of this project is the design and implementation of a platform that provides an online window towards *ecommerce* to small and medium local commerces that can not afford the development of their own. This platform must allow the search and following of commerces to the clients, and the consumption of the offers that the commerces create on the platform. It must also allow commerces the creation and consumption of offers, as well as client notification of new offers.

The platform's backend consists on an API REST, que implements the required business logic. This service is implemented using *.NET Core*. Regarding data persistence, the project uses SQL Server, a relational database. Both the API REST and the database are deployed in Azure services.

The frontend is separated in 2 mobile applications, *Trocos* and *TrocosComercios*, developed in Xamarin, using XAML and C#.

Trocos will be used by the clients, who will be able to search and follow the commerces that are registered in the system, as well as obtaining QR codes for the offers they create on the platform.

On the other side, *TrocosComercios* will be available to the commerces, where they will be able to consult statistics about their commerces, create new offers and scan clients' QR codes in order to consume the offers.

Palabras chave:

- Aplicación móbil
- .NET Core
- Xamarin
- SQL Server
- C#
- Azure Notifications Hub

Keywords:

- Mobile application
- .NET Core
- Xamarin
- SQL Server
- C#
- Azure Notifications Hub

Índice Xeral

Índice de Figuras

Índice de Táboas

Capítulo 1

Introdución

Nun mundo no que o comercio electrónico está en auxe, e segue gañando cada vez máis popularidade, todo aquel que non está nel, está ante unha gran desventaxa fronte aos seus competidores. E como non é habitualmente algo barato, o comercio local moitas veces non pode permitirse o luxo de crear unha plataforma online ou pagar por algún servizo de *ecommerce*.

As multinacionais teñen webs e aplicacións propias, onde ofertan os seus productos e ofrecen descontos a quenes a utilizan, dando moitas facilidades e mellores ofertas aos seus clientes, además de moitas veces ofrecerlles mellores ofertas canto máis utilicen as súas plataformas, conseguindo así unha aínda maior fidelización da súa clientela.

Por este motivo, as grandes empresas exercen aínda máis presión sobre o pequeno e mediano comercio. Neste contexto, xorde a idea de *Trocos*, unha plataforma onde os comercios poden ofertar os seus productos, e tamén para que os clientes poidan buscar os comercios que teñen ao seu arredor. Deste xeito, a plataforma é beneficiosa tanto para clientes como para comercios.

A idea é conseguir unha plataforma común onde os clientes podan descubrir o comercio local das súas localidades, e os comercios podan utilizar a plataforma para realizar ofertas e poder conseguir unha fidelización de clientes a través dunha aplicación móbil que lles da acceso online, de maneira que obteñan máis visibilidade.

1.1 Obxectivos

O obxectivo principal deste proxecto é crear unha plataforma móbil onde tanto clientes como comercios poderán interactuar de forma beneficiosa para ambos. Por un lado, os clientes poderán buscar e seguir os comercios que lles interesen, recibindo información específicamente dos comercios que decidan seguir.

En canto aos comercios, poderán ofertar os seus productos e enviarlle notificacións aos

clientes que sigan os seus comercios, de maneira que a publicidade que realicen será sobre unha clientela específica que os segue, de maneira que é moi probable que as ofertas lles resulten interesantes. Además, poderán consultar estadísticas sobre esa clientela e o tipo de ofertas que máis consumen.

Tamén se busca con este proxecto ampliar os coñecementos sobre o desenvolvemento de aplicacións móbiles e a xestión de notificacións, posto que é un mundo en auxe, nunha sociedade que cada vez se apoia máis nos seus dispositivos móbiles.

1.2 Visión global do sistema

O sistema consistirá en dúas aplicacións móbiles implementadas utilizando Xamarin, unha orientada para os clientes, que poderán interactuar cos comercios e as súas ofertas; e outra para os comercios, onde poderán dar de alta novas ofertas, enviarlle notificacións aos clientes dos seus comercios e consultar estadísticas sobre o seu comercio.

Ambas aplicacións consumirán os recursos ofrecidos por un API REST implementado en .NET Core, no que se implementará toda a lóxica de negocio necesaria para o funcionamento da plataforma.

Os datos persistiranse utilizando unha base de datos relacional, utilizando SQL Server.

A continuación, na figura 1.1, podemos ver un esquema da estrutura do sistema

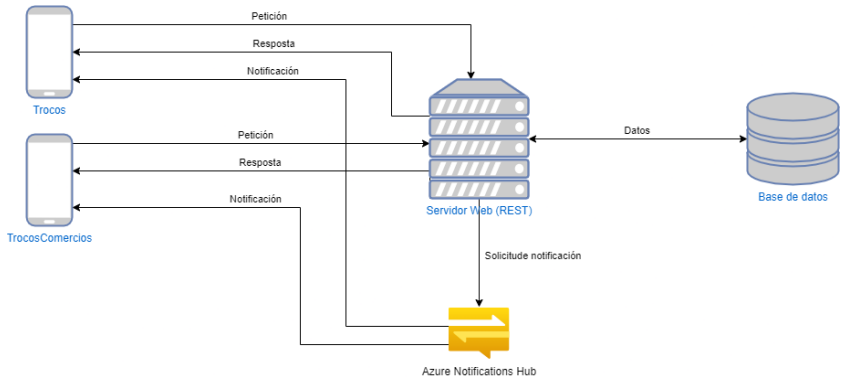

Figura 1.1: Arquitectura do sistema

Capítulo 2

Estado da arte

Hoxe en día existen ferramentas con funcionalidades e fines parecidos ao que se pretende conseguir con Trocos. Sen embargo moitas destas, ou ben están específicamente creadas para unha empresa particular, ou ben están ligadas a tarifas de pago con algunha entidade bancaria.

2.1 Alavuelta

Un exemplo recente desto é a nova aplicación de Abanca, Alavuelta, unha aplicación de fidelización de clientes e big data.

Trátase dunha aplicación móbil divida en tres servizos, dispoñibles para o comercio según o paquete que contrate. O máis básico (dispoñible a un menor prezo) permite consultar, comparar e interpretar información sobre os clientes e o negocio; mentres que os outros dous (ambos dispoñibles co outro paquete de contratación) serven para enviar notificacións a potenciales clientes e crear un escaparate online e tarxetas regalo e de fidelización.

2.2 Waylet

Waylet é unha aplicación móbil de Repsol. Esta aplicación permite acceder a ofertas exclusivas pagando directamente dende ela nas distintas estacións de servizo de Repsol e nalgúns establecementos e comercios (como El Corte Inglés).

Mediante o seu uso, o usuario pode ir acumulando saldo para utilizar nas súas compras e subindo de nivel para acceder a promocións aínda mellores.

Nesta aplicación podemos observar funcionalidades moi interesantes como o pago mediante o uso de QR e o sistema de niveles para gamificar a aplicación e motivar o seu uso. Por outra parte, é unha aplicación centrada nun comercio de grandes empresas como son Repsol e o corte inglés, o cal dista do obxetivo deste proxecto, que é o pequeno e medio comercio.

2.3 Burger King

Burger King posúe unha aplicación móbil baixo o seu mesmo nome na que publicita ofertas nos seus establecementos.

Esta aplicación é lixeiramente distinta, xa que está orientada tamén á realización de pedidos, tanto a domicilio como para recoller no establecemento, aínda que tamén comparte a funcionalidade de emisión de ofertas.

Dentro da aplicación, o usuario pode atopar ofertas exclusivas que pode trocar mediante o uso cos cupones que a plataforma proporciona.

Capítulo 3

Metodoloxía

No mundo do desenvolvemento de software, é necesaria a aplicación dunha metodoloxía de traballo para realizar unha correcta xestión e realización dun proxecto, xa que esta nos otorga unha estrutura e unha planificación máis que necesarias.

Para o desenvolvemento de Trocos, elexiuse unha metodoloxía áxil, debido a que é a práctica habitual na empresa, principalmente por ser un equipo pequeno e polos beneficios que aporta.

No caso concreto deste proxecto, esta metodoloxía está lixeiramente modificada debido a que o equipo está formado por unha soa persoa, o que fai innecesarios certos aspectos da metodoloxía habitual.

3.1 Scrum

Scrum [1] é unha métodoloxía áxil que recolle uha serie de boas prácticas para o traballo en equipo, buscando obter o mellor resultado no desenvolvemento dun proxecto.

A principal característica son as entregas iterativas, que consisten en entregas parciais do producto, posto que é unha metodoloxía deseñada para obter resultados pronto nun entorno complexo, onde os requisitos poden ser cambiantes ou están pouco definidos.

En Scrum divídese o traballo en ciclos temporais curtos e definidos, denominados *sprints*, dos que se produce como resultado un incremento do produto final que poida ser entregado ao cliente cando así o necesite.

Scrum tamén define unha serie de roles, pero como para o caso deste proxecto o equipo está conformado por unha soa persoa, estos están mesturados para adecuarse ás necesidades concretas.

3.1.1 Roles

En Scrum defínense unha serie de roles que recollen as competencias necesarias para poder levar a cabo o proxecto.

Estes roles son os seguintes:

- Product Owner: É o responsable da representación do cliente ou clientes, encargado de maximizar o valor do produto desenvolto polo equipo scrum, e de xestionar a carteira de requisitos, tanto a súa definición como a súa valoración.
- Scrum Master: É o individuo que traballa tanto co equipo como co Product Owner, asegurándose de que o traballo segue os cauces establecidos. É o encargado de asistir ao PO na xestión da carteira de requisitos, ou Backlog, e de organizar os sprints e orientar ao equipo de desenvolvemento na realización das súas tarefas. Debe ser unha persoa con coñecementos avanzados en Scrum e debe estar implicado no proxecto, a poder ser tendo gran coñecemento del e buscando sempre a optimización da posta en práctica de dita metodoloxía.
- Equipo de desenvolvemento: Son o equipo de desenvolvemento encargado de implementar o especificado nas distintas tarefas definidas para cada sprint.

3.1.2 Sprints

En Scrum, os proxectos repártense en bloques temporais que se recomenda que duren entre unha e catro semanas.

Para cada unha destas iteracións, establécense unha lista de tarefas, acordadas entre o equipo e os clientes, obtendo ao rematar o *sprint* un incremento do producto, ou sexa, un resultado con valor para o cliente.

Durante cada iteración, realízanse as tarefas de selección de requisitos e planificación, deseño e implementación do produto, e proba e revisión do traballo realizado durante dito *sprint*.

Tamén se realizan reunións diarias de sincronización para manter a comunicación entre os membros do equipo e poder facer adaptacións no caso de ser necesarias para cumplir cos prazos establecidos.

3.1.3 Artefactos

Scrum define unha serie de artefactos ou elementos físicos que se producirán como resultado da súa aplicación. Son os seguintes:

- Product Backlog: Inventario que recolle as tarefas, casos de uso e requisitos do proxecto. Constitúe a principal fonde de información do produto, e é o resultado da colaboración do Product Owner co cliente ou clientes e o Scrum Master. Esta lista irase actualizando conforme vaia avanzando a realización do proxecto e vaian aparecendo ou cambiando novos requerimentos e prioridades.

- Sprint Backlog: Conxunto de tarefas que saen do Product Backlog para ser abordadas durante un *sprint*. O equipo de desenvolvemento é o encargado da xestión e estimación destas tarefas.

- Incremento: O resultado dun *sprint*, é dicir, a suma de tódolos elementos desenvoltos e que será posta a disposición do usuario final en forma de software.

- Historias de usuario: Técnica de toma de requisitos que consiste en definir a funcionalidade de forma breve e utilizando unha linguaxe de expresión común, que sexa entendible para todo o mundo.

3.2 Aplicación de Scrum

Debido a que o equipo está composto por unha soa persoa, e non existe un cliente como tal, xa que se trata dunha idea interna da empresa, a metodoloxía empregada na realización deste proxecto recolle certas características de Scrum pero adaptadas á casuística concreta.

3.2.1 Roles

Posto que so se dispón de unha persoa e non hai cliente, os tres roles quedan condensados principalmente no alumno, que aínda así comparte certas responsabilidades do Product Owner con ambos titores do proxecto, quenes aportaron ideas e opinións sobre o produto a desenvolver, ademáis de axudar a definir certos aspectos do Product Backlog.

Por outro lado, o alumno adoptou os roles de Scrum Master e do equipo de desenvolvemento, encargándose de organizar o traballo a desenvolver nos distintos sprints e de estimar a duración das distintas tarefas.

3.2.2 Artefactos

Para levar un correcto seguimento e organización dos artefactos creados coa intención de xestionar o proxecto, utilizouse a ferramenta Redmine [2], que permite a creación de tarefas, a imputación de horas e a definición de distintos estados polos que pode pasar.

Para cada *sprint*, fóronse seleccionando tarefas do Product Backlog e marcándose con etiquetas para identificar a cal pertencían, e foise actualizando o seu estado según se ían realizando.

3.2.3 Reunións

Debido ao caso especial de ter un so integrante no equipo do proxecto, antes de cada *sprint*, o alumno dedicaba parte do tempo en organizar as tarefas que se ían incluír, así como revisar o Product Backlog e actualizalo no caso de ser necesario. Tra-la realización deste traballo, pasáballe un resumo ao titor para que dese a súa aprobación e procedíase á realización do *sprint*.

Periódicamente fixéronse reunións co titor para amosar o traballo realizado e discutir o camiño a seguir, descubrindo así novos requisitos ou adaptando os xa existentes.

Capítulo 4

Análise de requistos global

Detallaranse a continuación os distintos roles de usuario que terán acceso ao sistema, así como os requisitos globais do mesmo.

Esta recolección de requisitos realizouse utilizando a ténica de historias de usuario, tal como aconsella Scrum.

4.1 Roles

Os actores deste proxecto diferéncianse entre sí polos permisos que posúen cando acceden á aplicación, polo que distinguimos tres roles distintos:

- Usuario sen autenticar: Os usuarios que non inicien sesión no sistema so poderán rexistrarse ou iniciar sesión.
- Cliente autenticado: Definimos como cliente a aquel usuario identificado mediante un usuario e un contrasinal que non posúa un comercio. Estes usuarios poderán realizar búsquedas de comercios, seguilos se así o desexan, e consumir as súas ofertas se son seguidores de dito comercio e posúen o nivel suficiente.
- Comerciante autenticado: Referímonos a comerciante a aquel usuario identificado mediante un usuario e un contrasinal que teña asociado un comercio a súa conta. Además das funcionalidades de un cliente autenticado, un comerciante poderá dar de alta ofertas e consultar datos sobre o seu comercio, así como escanear códigos de ofertas para que os clientes podan consumilas.

4.2 Historias de usuario

Detállanse a continuación as historias de usuario que constitúen o Product Backlog do proxecto.

Estas recollen nunha linguaxe coloquial unha funcionalidade específica. Adicionalmente, inclúese xunto a elas unha descrición aclarativa do seu obxetivo e algúns mockups, para ampliar a información que ofrecen.

4.2.1 Product backlog

ID	Nome	Descrición
US01	Rexistro	Como cliente non autenticado quero rexistrarme no sistema
US02	Autenticación cliente	Como cliente non autenticado quero iniciar sesión no sistema
US03	Autenticación comerciante	Como comerciante non autenticado quero iniciar sesión no sistema
US04	Pechar sesión como cliente	Como cliente autenticado quero pechar sesión no sistema
US05	Pechar sesión como comerciante	Como comerciante autenticado quero pechar sesión no sistema
US06	Alta de oferta	Como comerciante autenticado quero dar de alta unha oferta
US07	Visualizar comercio	Como comerciante autenticado quero ver os datos do meu comercio
US08	Escanear oferta	Como comerciante autenticado quero escanear un código QR dun cliente
US09	Consultar estadísticas	Como comerciante autenticado quero consultar estadísticas do meu comercio
US10	Ver puntos e categoría	Como cliente autenticado quero ver os meus puntos e categoría
US11	Buscar comercio	Como cliente autenticado quero buscar un comercio buscando por distintos parámetros
US12	Ver mapa	Como cliente autenticado quero visualizar os comercios no mapa
US13	Ver comercios	Como cliente autenticado quero visualizar os comercios que sigo
US14	Ver detalles de comercio	Como cliente autenticado quero visualizar os detalles dun comercio
US15	Seguir comercio	Como cliente autenticado quero seguir a un comercio
US16	Deixar de seguir comercio	Como cliente autenticado quero deixar de seguir a un comercio
US17	Ver ofertas	Como cliente autenticado quero visualizar as ofertas dispoñibles para min
US18	Ver detalle de oferta	Como cliente autenticado quero visualizar os detalles dunha oferta dispoñible para min
US19	Ver QR de oferta	Como cliente autenticado quero visualizar o código QR dunha oferta para poder consumila

Táboa 4.1: Product Backlog

4.2.2 Funcionalidades

US01 - Rexistro

Para darse de alta no sistema, os usuarios non autenticados deberán introducir un nome de usuario e un contrasinal. En caso de algún dos campos introducidos non sexa correcto, o sistema amosará unha mensaxe indicando o problema. No caso contrato, o cliente quedará autenticado.

US02 - Autenticación cliente

Para autenticarse no sistema, os clientes deberán introducir o seu nome de usuario e o contrasinal. En caso de ser correctos, os clientes estarán autenticados e terán acceso ás funcionalidades dispoñibles para un cliente autenticado.

No caso de que algún dos campos non sexa correcto, o sistema amosará unha mensaxe indicando o problema.

US03 - Autenticación comerciante

Para autenticarse no sistema, os comerciantes deberán introducir o seu nome de usuario e o contrasinal. En caso de ser correctos e de ter algún comercio asociado a súa conta, os comerciantes estará autenticado e terá acceso ás funcionalidades dispoñibles para un comerciante autenticado.

No caso de que algún dos campos non sexa correcto, o sistema amosará unha mensaxe indicando o problema.

US04 - Pechar sesión cliente

Os clientes autenticados poderán pechar a sesión sempre que o desexen, premendo no botón *Cerrar sesión*. A partir dese momento, só poderán acceder ás funcionalidades dispoñibles para usuarios non autenticados e deixarán de recibir notificacións da aplicación.

US05 - Pechar sesión comerciante

Os comerciantes autenticados poderán pechar a sesión sempre que o desexen, premendo no botón *Cerrar sesión*. A partir dese momento, só poderán acceder ás funcionalidades dispoñibles para usuarios non autenticados.

US06 - Alta de oferta

Os comerciantes autenticados poderán dar de alta ofertas para os seus comercios pulsando en *Alta de oferta*.

Para facelo, deberán cubrir un formulario cos seguintes campos:

- Descrición: Descrición aclarativa de en qué consiste a nova oferta.
- Categoría: Categoría de usuarios que terán acceso a esta oferta.
- Data de inicio: Data a partir da que estará dispoñible a oferta.
- Data de fin: Data a partir da que deixará de estar dispoñible a oferta.

Unha vez introducidos os campos, o sistema realizará unha validación e amosará unha mensaxe de erro no caso de existir algún. Se non, quedará dada de alta no sistema e enviarase unha notificación aos seguidores do seu comercio. A continuación, na figura 4.1 podemos ver un mockup de como sería a ventá.

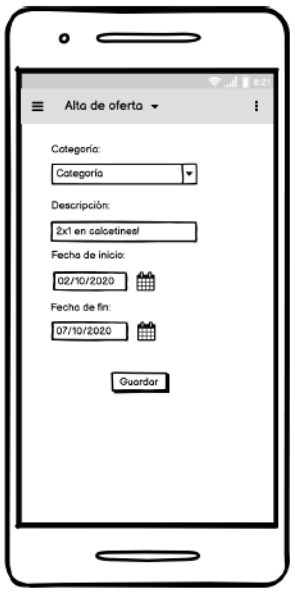

Figura 4.1: Mockup da vista de alta de oferta

US07 - Visualizar comercio

Os comerciantes autenticados poderán ver os datos do seu comercio pulsando en *Mi Comercio*.

Verán o nome, o tipo, o CIF, a provincia e a poboación do seu comercio, así como as ofertas dispoñibles e as bloqueadas (ben porque a data de inicio é posterior ao día da consulta, ou porque a data de fin é anterior).

US08 - Escanear oferta

Os comerciantes autenticados poderán escanear unha oferta seleccionando a opción *Escanear*.

Nese momento, a aplicación utilizará a cámara do seu dispositivo móbil para ler o código QR desexado. Unha vez lido, amosará unha mensaxe de confirmación e enviará unha notificación ao cliente cuxa oferta acaba de ser consumida de que a transacción rematou con éxito; ou ben amosará unha mensaxe de erro indicando o problema.

US09 - Consultar estadísticas

Os comerciantes autenticados poderá acceder a unha serie de estadísticas do seu comercio seleccionando a opción de *Estadísticas*.

O sistema amosará unha pantalla na que se poderán visualizar os seguidores do comercio, cantas ofertas se consumiron, e de que categorías son.

US10 - Ver puntos e categoría

Os clientes autenticados poderán ver os puntos que teñen actualmente na aplicación e a que categoría pertencen na páxina de *Inicio*.

Dita información estará amosada na pantalla principal da aplicación cunha cor asociada a súa categoría e unha barra de progreso que indicará cantos puntos necesita para alcanzar o seguinte nivel.

US11 - Buscar comercio

Os clientes autenticados poderán realizar unha búsqueda de comercios seleccionando a opción *Búsqueda*.

Nese momento, o sistema amosará un elemento para introducir texto no que o cliente poderá introducir a cadea de texto que desexe e o sistema amosará os comercios que conteñan dita cadea nalgunha das súas propiedades.

US12 - Ver mapa

Os clientes autenticados poderán seleccionar a opción *Mapa* para visualizar un mapa cos comercios situados nel.

O sistema solicitará permiso para utilizar a localización do usuario. No caso de rexeitar, o mapa amosarase sen a localización do cliente. No caso de aceptar, o cliente poderá ver a súa situación a tempo real no mapa, así como ver a localización dos distintos comercios, podendo seleccionalos para ir ao seu detalle.

US13 - Ver comercios

Os clientes autenticados poderán seleccionar a opción de *Mis Comercios*. Nela poderán ver a lista de comercios aos que seguen.

US14 - Ver detalles de comercio

Se os clientes autenticados seleccionan un comercio no mapa, na lista de comercios aos que seguen, ou nunha búsqueda, poderán acceder aos detalles do comercio, onde verán infor-

mación extendida sobre o comercio, así como a lista de ofertas ás que teñen acceso, e a lista de ofertas que teñen bloqueadas por non ter a categoría suficiente.

Podemos observar un *mockup* deste US na figura 4.2.

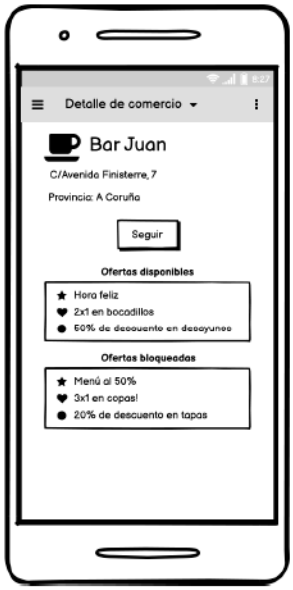

Figura 4.2: Mockup da vista de detalles de comercio

US15 - Seguir comercio

Dende a pantalla de búsqueda ou o dende os detalle de comercio, os clientes autenticados poderán seguir ao comercio. Se deciden facelo, pasarán a ter acceso ás ofertas de dito establecemento e recibirán notificacións de cando ese comercio publique unha nova oferta.

US16 - Deixar de seguir comercio

Dende a pantalla de búsqueda, dende a dos comercios aos que segue ou dende os detalle de comercio, os clientes autenticados poderán deixar de seguir a un comercio. Se deciden facelo, deixarán de ter acceso ás ofertas de dito establecemento e deixarán de recibir notificacións das novas ofertas del.

US17 - Ver ofertas

Na pantalla inicial da aplicación, os clientes autenticados poderán ver o listado de ofertas que teñen dispoñibles debido á súa categoría, dos distintos comercios aos que seguen.

Na figura 4.3 podemos apreciar o bosquexo desta ventá.

Figura 4.3: Mockup da vista de inicio

US18 - Ver detalle de oferta

Se os clientes autenticados seleccionan unha oferta do listado de ofertas da pantalla inicial, ou ben dende o listado de ofertas dispoñiblos da vista de detalle dun comercio, pasarán a unha pantalla onde poderán ver información extendida dunha oferta, así como unha imaxe a pequena escala do código QR que se utiliza para consumir unha oferta.

US19 - Ver QR de oferta

Se os clientes autenticados pulsan na imaxe a pequena escala do QR dende os detalles dunha oferta, amosarase o código QR ampliado para facilitar a súa lectura por parte do comerciante. A continuación, na figura 4.4 podemos ver o esquema da ventá

Figura 4.4: Mockup da vista de detalle de oferta

Capítulo 5

Planificación

Neste capítulo abórdase a planificación do proxecto, realizada seguindo as bases da metodoloxía *Scrum*.

Apórtase a continuación o detalle de cada un dos sprints realizados durante o proxecto, falando do obxectivo de cada un e do seu *Sprint backlog*.

5.1 Sprints

Detallaranse agora os sprints realizados durante este proxecto, que se foron planificando progresivamente a medida que remataba un e antes de empezar o outro, mediante a planificación do alumno e ca aprobación do titor.

5.1.1 Sprint 0: Concepción do producto

No sprint inicial realizáronse unha serie de reunións para compartir ideas e enfocar o proxecto, dictaminando que ruta tomar, pensando nas funcionalidades que se ían implementar, en qué tecnoloxías e en qué plataformas.

5.1.2 Sprint 1: Estruturación

O obxectivo desta iteración é instalar tódalas ferramentas necesarias e desenvolver arquetipos inciais para ambas aplicacións móbiles, así como un backend con endpoints de exemplo para verificar que unha vez desplegado no servidor de desenvolvemento este é accesible.

5.1.3 Sprint 2: Xestión de básica usuarios

Neste *Sprint* abórdase todo o relacionado cos usuarios, concretamente as historias:

- US01 - Rexistro

- US02 - Autenticación cliente
- US03 - Autenticación comerciante
- US04 - Pechar sesión como cliente
- US05 - Pechar sesión como comerciante
- US10 - Ver puntos e categoría

5.1.4 Sprint 3: Búsqueda e visualización de comercios

Impleméntanse as funcións que fan posible a navegación polos distintos comercios da aplicación:

- US07 - Visualizar comercio
- US11 - Buscar comercio
- US13 - Ver comercios
- US14 - Ver detalles de comercio
- US15 - Seguir comercio
- US16 - Deixar de seguir comercio

5.1.5 Sprint 4: Xestión de ofertas

Despois de facer posible a visualización de comercios, pasamos ás ofertas, posto que dependen dos
comercios para existir:

- US06 - Alta de oferta
- US17 - Ver ofertas
- US18 - Ver detalle de oferta

5.1.6 Sprint 5: Implementación de QRs, consumición de ofertas e estadísticas

Neste *sprint* trátase a creación das imaxes cos códigos QR e a funcionalidade de poder consumir ofertas escaneando ditos códigos. Tamén se inclúe a historia de ver as estadísticas dun comercio:

- US19 - Ver QR de oferta
- US09 - Consultar estadísticas
- US08 - Escanear oferta

5.1.7 Sprint 6: Xestión de notificacións

Unha vez implementada as funcionalidades de consumir ofertas e altas de ofera, podemos desenvolver o centro de notificacións.

5.1.8 Sprint 7: Conexión ca API de Google e mapa

Neste *sprint* abordamos a visualización do mapa:

- US12 - Ver mapa

5.1.9 Sprint 8: Elaboración da memoria

Na última iteración realízase a elaboración desta memoria. Tamén se inclúen melloras e a solución dalgún problema, detectados na última reunión de revisión do anterior sprint.

5.2 Planificación temporal

Sprint	Nome	Inicio	Fin	Horas
0	Concepción do producto	23/06/20	11/07/20	43
1	Estruturación	12/07/20	08/08/20	92
2	Xestión básica de usuarios	09/08/20	15/09/20	127
3	Búsqueda e visualización de comercios	16/09/20	08/10/20	114
4	Xestión de ofertas	11/10/20	05/11/20	129
5	Implementación de QRs, consumición de ofertas e estadísticas	06/11/20	20/12/20	93
6	Xestión de notificacións	21/12/20	16/01/21	87
7	Conexión ca API de Google e mapa	17/01/21	31/01/21	54
8	Elaboración da memoria	01/02/21	22/02/21	98
TOTAL		23/06/20	22/02/21	837

Táboa 5.1: Planificación temporal e horas adicadas

Capítulo 6

Fundamentos tecnolóxicos

Neste capítulo explicaranse as ferramentas utilizadas para levar a cabo este proxecto. Farase unha división en bloques según onde foron empregadas para facilita-la súa lectura. Son os seguintes:

- Tecnoloxías empregadas no *backend*
- Tecnoloxías empregadas no *frontend*
- Tecnoloxías complementarias

6.1 Tecnoloxías empregadas no *backend*

O *backend* está conformado por unha *API REST* desenvolvida en Net Core [3]. Seguidamente detallaranse as ferramentas empregadas para a súa realización.

6.1.1 SQL Server

SQL Server [4] é un sistema de xestión de base de datos relacional, desenvolvido por Microsoft. É un sistema completo, que inclúe características interesantes, como:

- Transaccións, que nos permiten facer unha serie de actualizacións, insercións ou borrados de forma consistente.
- Procedementos almacenados, útiles para a programación de consultas complexas ou tarefas automáticas para o mantemento da base de datos.
- Traballo en modo cliente-servidor, o cal é moi útil para este proxecto xa que a base de datos atópase aloxada nos servidores de Azure [5].

As principais razóns polas que se optou por este tipo de base datos é a experiencia previa da empresa con este tipo de tecnoloxía, e a facilidade que ten para ser desplegada nos servidores de Azure, que é onde se atopa todo o *backend* da aplicación.

6.1.2 C#

C# [6] é unha linguaxe de programación baseada en obxetos e con seguridade de tipos, que ten as súas raíces na familia de linguaxes C, e que permite desenvolver aplicacións executadas no sistema de .NET. Como é a linguaxe que utiliza Xamarin [7], a tecnoloxía ca que se desenvolve o *frontend*, optouse por utilizalo tamén no *backend* para facilitar a transmisión de información, sobre todo do modelo da aplicación.

6.1.3 .NET

.NET é unha plataforma de desenvolvemento gratuita e de código aberto. Ten ferramentas para crear moitos tipos distintos de aplicacións, entre elas un API web REST, que é o caso deste proxecto.

6.1.4 Entity Framework

Entity Framework [8] trátase dun conxunto de tecnoloxías orientadas a datos. Permiten modelar entidades e relacións dunha base de datos e relacionalas fácilmente con obxetos, facilitando moito esta conexión. Neste proxecto utilízouse moito para as distintas operacións de recuperación de datos que se debían levar a cabo, posto que moitas das entidades están fortemente relacionadas.

6.1.5 Rest

REST (*REpresenational State Transfer*) é un estilo de arquitectura baseado nas características da web, que serve para desenvolver aplicacións distribuídas. Normalmente unha aplicación REST expón unha serie de recursos (que se transmiten mediante JSON ou XML) e un conxunto de operacións para traballar con eles.

6.1.6 Visual Studio

Microsoft Visual Studio [9] é un entorno de desenvolvemento integrado (IDE) compatible con moitas linguaxes de programación e entornos de desenvolvemento web, ademáis da conectividade da que dispón con Windows Azure, o cal foi unha cualidade moi utilizada durante o desenvolvemento deste proxecto.

6.2 Tecnoloxías empregadas no *frontend*

Falarase a continuación das tecnoloxías empregadas para desenvolver o *frontend*.

6.2.1 Xamarin

Xamarin é un framework de código aberto para o desenvolvemento multiplatarforma (iOS, Android e Windows con .NET). Esta plataforma aporta unha capa de abstracción que se encarga da comunicación do código compartido co da plataforma subxacente. Xamarin execútase nun entorno administrado, o que permite a realización da recolección de elementos non utilizados e a asignación de memoria.

A gran vantaxe disto é a capacidade que aporta aos programadores de escribir toda a lóxica de negocio prácticamente nunha soa linguaxe (ata o 90% do código pode ser compartido), pero obter unha aplicación e apariencia nativos de diversas platarformas.

As aplicacións pódense escribir en PC o Mac, utilizando C# e as BCL (Biblioteca de clases base) de .NET, que consiste nunha gran colección de clases con diversas características, entre elas conexión con bases de datos, serialización, etc.

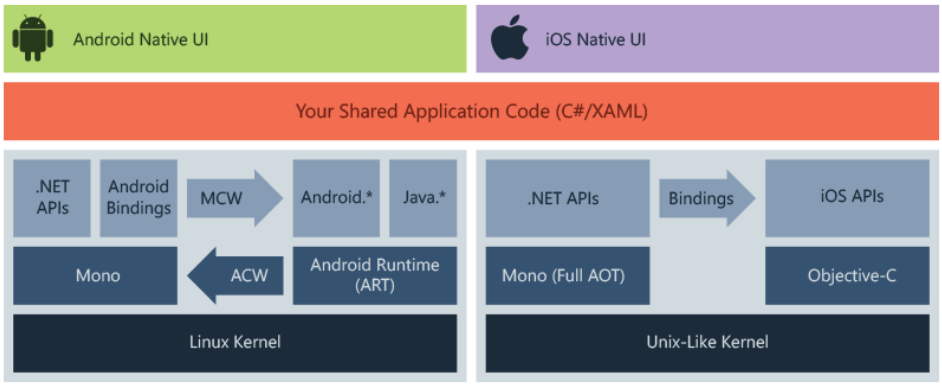

Figura 6.1: Funcionamento de Xamarin

6.2.2 Azure Notifications Hub

Azure Notification Hubs [10] é unha plataforma de notificacións móbiles que permite enviar un gran volume de notificacións tanto a dispositivos iOS, Android, Windows e Kindle, que funcionen con APNs (Apple Push Notification Service), FCM (Firebase Cloud Messaging), MPNS (Servicio de notificaciones push de Microsoft)..., dunha forma masiva e escalable. Cunha configuración mínima, grazas ao seu sistema de etiquetas, pódense adaptar as notificacións para enviarse masivamente ou a clientes específicos.

As principais razóns polas que se escolleu este sistema de notificacións foi a súa alta compatibilidade con Xamarin (grazas ás librerías publicadas en NuGet [11]) e o sistema de etiquetas, que aporta moita flexibilidade á hora de enviar notificacións a grupos concretos de dispositivos.

6.2.3 Firebase Cloud Messaging

Firebase Cloud Messaging [12] é unha solución de mensaxería multiplataforma que permite enviar notificacións de forma segura. É o sistema que permite a Azure Notifications Hub enviar notificacións a dispositivos Android.

Por esta razón, utilizouse neste proxecto para actuar de ponte entre o Azure Notifications Hub e os dispositivos Android nos que se instala a nosa aplicación.

6.3 Tecnoloxías complementarias ao desenvolvemento

6.3.1 Git

Git [13] é un sistema de control de versións distribuído gratuíto e de código aberto de gran versatilidade, que permite xestionar dende pequenos a grandes proxectos.

Diferénciase doutros SCM por ser distribuído, permitindo ter a cada usuario unha copia local do repositorio sobre a que realizar commits e demáis operacións. Ademáis posúe ferramentas para o tratamento de ramas e xestión de diferenzas entre arquivos de gran eficiencia, convertíndoo nun dos mellores e máis utilizados SCM.

6.3.2 Github

Github [14] é unha plataforma para aloxar proxectos que utiliza o sistema Git que fomenta o software libre e permite repositorios públicos e privados de forma gratuíta.

6.3.3 Redmine

Redmine é unha ferramenta para a xestión de proxectos, que permite, mediante diversas funcionalidades, realizar o seguimento e organización dos mesmos.

Unha das características que máis se utilizaron neste proxecto foi o sistema de tarefas e a actualización de estados destas. Utilizáronse para ir definindo as distintas tarefas que foron surxindo das historias de usuario, planificándose antes de cada sprint e imputando as horas de traballo realizadas sobre cada unha.

Capítulo 7

Desenvolvemento

Neste capítulo falarase dos aspectos máis importantes do desenvolvemento da aplicación. Trataranse tanto termos técnicos como funcionais que foron surxindo durante as diversas iteracións.

7.1 Modelo de datos

Na figura 7.1 podemos visualizar o modelo de datos de Trocos, que consta de 3 unidades principais, *Customer*, *Commerce* e *Offer*. Un comercio so pode existir se hai un cliente que o posea, e unha oferta so pode existir se hai un comercio que a oferte.

Un cliente pode consumir ofertas, e as ofertas poden ser consumidas por varios clientes. Así mesmo, un cliente pode seguir a comercios, e os comercios poden ser seguidos por varios clientes.

Por outro lado, temos os enumerado *Category*, que serve para establecer as categorías (niveis) das ofertas e clientes e a súa cor dentro da aplicación, e o enumarado *CommerceType*, que establece o tipo de comercio e o seu icono.

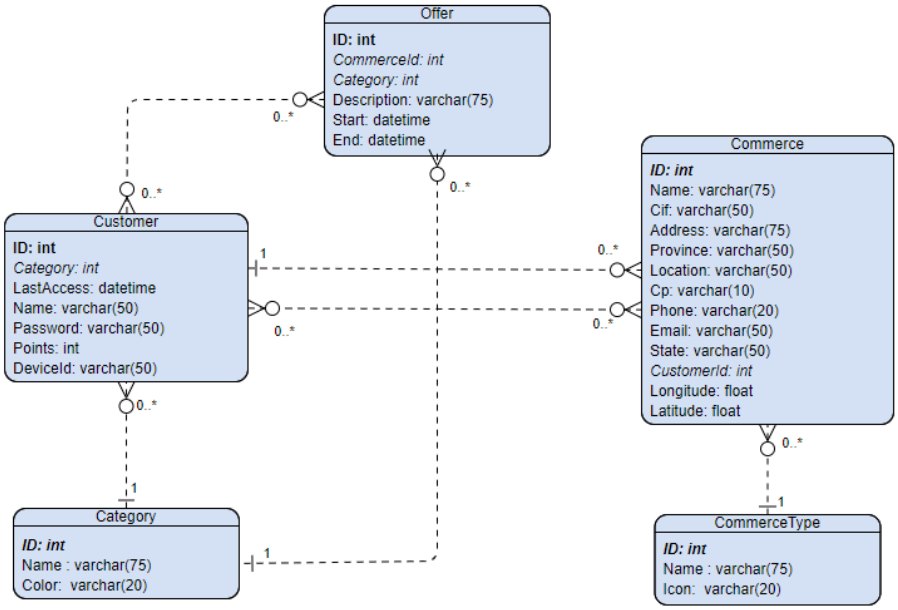

Figura 7.1: Diagrama de entidades da aplicación

7.2 Sprint 0: Concepción do producto

No sprint inicial leváronse a cabo unha serie de reunións informais nas que se discutiu a idea de Trocos e as mellores formas de levala a cabo, estudando cales serían as súas funcionalidades e obxectivos.

A idea era crear unha plataforma na que os usuarios puidesen descubrir os comercios locais e ter acceso a ofertas por utilizar a aplicación, dando deste xeito a ditos comercios máis visibilidade online fronte ás grandes empresas que se poden permitir ter aplicacións e plataformas propias.

A decisión inicial foi optar pola plataforma móbil, xa que se considerou a máis viable para as transaccións entre cliente e comerciante. Isto tamén implicou separar Trocos en dúas aplicacións, unha para os comerciantes e outra para os clientes, posto que as funcionalidades estaban bastante separadas e non tería sentido unificalas nunha soa; aínda que o backend sería común para ambas.

Estudando aplicacións similares, decidiuse que o mellor sistema para o intercambio de información entre cliente e comerciante serían os códigos QR, posto que son fáciles de manexar e fan que o intercambio de información sexa rápido e fácil de utilizar. Además, co código QR temos a posibilidade de incluír códigos firmados e cifrados para evitar brechas de seguridade, como poderían ser códigos QR xerados fora da aplicación para acceder a ofertas de outros

clientes.

Optouse tamén por gamificar lixeiramente a aplicación, incluíndo un sistema de categorías e puntos, o que pretende incentivar aos clientes a consumir ofertas mediante a aplicación, para ter acceso ás de categorías superiores.

7.3 Sprint 1: Estruturación

No segundo *sprint*, buscouse lograr a configuración dos entornos de desenvolvemento e lograr unha proba funcional con funcionalidades de proba tanto do *frontend* como do *backend*, para establecer así unha base da que partir.

7.3.1 Estrutura do *backend*

O *backend* creouse utilizando a ferramenta de creación de proxectos de Visual Studio, seleccionando a plantilla de proxecto de Aplicación Web ASP.NET Core. Seguíuse unha distribución MVC (Model View Controller), polo que se crearon as seguintes carpetas:

- Controllers: Dentro deste paquete irán tódolos controladores para as distintas entidades.
- Model: No modelo irán as clases das entidades que mapean cas táboas da base de datos, así como a clase WebApiFidCliContext, que é a clase na que define todo o contexto da base de datos, é dicir, as relacións entre entidades, restriccións de tipos, e a propia cadea de conexión.
- Service: Esta carpeta conterá os distintos servizos de cada unha das entidades, e implementarán a lóxica de negocio necesaria.

A plantilla traía un controller de exemplo chamado *WeatherForecastController*, que simula unha petición GET a unha api dunha estación meteorolóxica. Dito controller utilizouse para verificar que o despliegue nos servidores de Azure era correcto.

Para o despliegue, utilizamos a ferramenta de *Publicar...* de Visual Studio. Primeir, introducimos a nosa conta de Azure. Unha vez inciada a sesión, seleccionamos Azure, e despois Azure App Service (Windows). Seguidamente seleccionamos o noso tipo de suscrición, e dentro do grupo de recursos Trocos, que creamos previamente no portal de Azure, seleccionamos Trocos Api, o recurso de WebApi que creamos tamén no portal dentro do grupo de Recursos, e pulsamos finalizar. Unha vez feito esto, podemos pulsar en publicar, o que compilará e desplegará a nosa aplicación na URL indicada, neste caso http://trocosapi.azurewebsites.net. Cando remata o despliegue, probamos a URL https://trocosapi.azurewebsites.net/weatherforecast e verificamos que nos devolve o JSON cos datos de exemplo.

7.3.2 Estrutura do *frontend*

Para o *frontend* tomouse a decisión de separalo en dúas aplicacións móbiles, unha para os clientes (Trocos) e outra para os comercios (TrocosComercios). Para crealas, utilizamos a plantilla de proxecto de *Aplicación Móvil (Xamarin.Forms)* que nos ofrece Visual Studio. Este proxecto ven con 4 solucións. Unha é a principal, que contén as clases que son comúns ao resto de solucións, e logo trae unha para Android, outra para iOS e outra para UWP. Nestas 3 é onde se inclúe o código específico de cada plataforma, que se intentará que sexa o mínimo para facilitar o desenvolvemento multi-plataforma.

Seguirase en ambos proxectos o patrón MVVM (Model-View-ViewModel) para desacoplar o máximo posible a interfaz de usuario da lóxica da aplicación.

Na solución principal, créase a seguinte estrutura de carpetas:

- Exceptions: Onde se incluirán as excepcións propias da aplicación o caso de habelas.
- Models: Aquí irán as clases das entidades da nosa aplicación e o cliente que se encargará de chamar ao backend para recuperar ditas entidades, así como algunha clase máis que poida ser necesaria para o traspaso de información co API REST.
- Resources: Engadiranse aquí todos los arquivos de recursos que se utilicen para a aplicación, xa sexan imaxes, fontes, etc.
- Services: Esta carpeta conterá os servizos que poida ser necesario implementar para ser utilizados despois nos ViewModels.
- ViewModels: Nesta carpeta recolleranse todos os ViewModels das distintas vistas.
- Views: Aquí irán as clases que conterán as distintas vistas que conformen as aplicacións.

O proxecto tamén inclúe o ficheiro App.xaml, que contén o diccionario de recursos, onde podemos dar de alta estilos globais que se vaian a utilizar en todo o proxecto; o ficheiro AssemblyInfo.cs, onde se poden incluír instruccións de exportación de recursos (como fontes); o Bootstrap.cs, onde se definen os servizos do ServiceContainer, e o Config.cs, onde se poden definir variables globais.

7.4 Sprint 2: Xestión de usuarios

7.4.1 Análise

Como no anterior *sprint* quedou lista a base para o proxecto, neste seguinte decídese empezar cas historias de usuario do *Product Backlog*. Decídese comezar pola xestión de usuarios, que se amosa descomposta en tarefas nas tábas 7.1, 7.2, 7.3, 7.4 e 7.5:

US01 - Rexistro
Como usuario non autenticado quero rexistrarme no sistema.
Tarefas
Creación dun endpoint no *backend* que implemente unha operación de rexistro de cliente.
Creación da vista en Xamarin que conteña o formulario de rexistro.

Táboa 7.1: Sprint 1: Rexistro

US02 - Autenticación cliente
Como usuario non autenticado quero iniciar sesión no sistema.
Tarefas
Creación dun endpoint no *backend* que valide o usuario e contraseña e devolva un token jwt.
Creación da vista en Xamarin que conteña o formulario de login.

Táboa 7.2: Sprint 1: Autenticación cliente

US03 - Autenticación comerciante
Como comerciante non autenticado quero iniciar sesión no sistema.
Tarefas
Creación dun endpoint no *backend* que valide o usuario e contraseña e devolva un token jwt.
Creación da vista en Xamarin que conteña o formulario de login.

Táboa 7.3: Sprint 1: Autenticación comerciante

US04 - Pechar sesión como cliente
Como cliente autenticado quero pechar sesión no sistema.
Tarefas
Engadir o botón de *Cerrar sesión* ao menú da aplicación de clientes.

Táboa 7.4: Sprint 1: Pechar sesión como cliente

US05 - Pechar sesión como comerciante
Como comerciante autenticado quero pechar sesión no sistema.
Tarefas
Engadir o botón de *Cerrar sesión* ao menú da aplicación de comerciantes.

Táboa 7.5: Sprint 1: Pechar sesión como comerciante

7.4.2 Deseño e implementación

US01 - Rexistro

O rexistro de usuarios realízase mediante unha petición POST ao *backend* que contén os datos do usuario, que este introduciu previamente nun formulario no seu dispositivo móbil.

Para o mantemento da sesión, utilízanse os token JWT (JSON Web Token), xa permiten que o *backend* sexa totalmente stateless e además estea protexido de peticións non desexadas. Estes token están firmados mediante o algoritmo hash 256, e teñen caducidade dun día.

O endpoint de login devolveranos o token no caso de que as credenciais sexan correctas, e despois a aplicación cliente deberá encargarse de enviar dito token en cada petición que realice ao *backend*, para manterse autorizado. No caso de que unha petición non conteña o token, ou este sexa incorrecto ou esté caducado, o backend responderá cun código 401 (*Unauthorized*).

En NuGet temos a librería JwtBearer, que nos aporta as ferramentas necesarias para xerar un token *JWT* e poder engadir os claims que consideremos necesarios. Neste caso, almacénase o login de usuario. Na figura 7.2 podemos ver un pequeno diagrama de flujo de como funcionan as peticións con autorización mediante tokens *JWT*

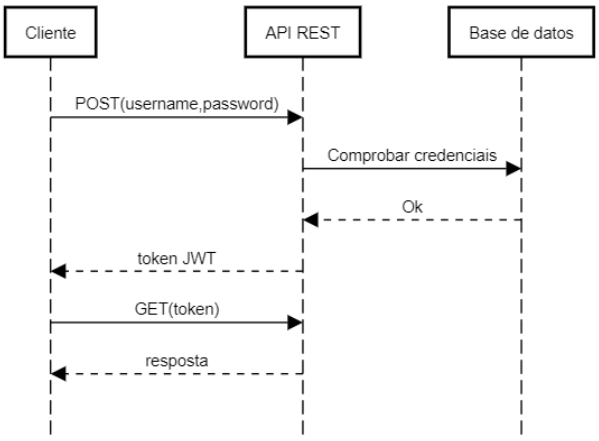

Figura 7.2: Diagrama de secuencia token JWT

Para este caso, realízanse validacións a nivel de cliente, e validacións a nivel de servidor, xa que deben comprobarse que os valores introducidos son correctos, pero que además non se produzan duplicidades de usuario na base de datos. En ambos casos, amósase unha mensaxe de erro indicando o problema.

Se o rexistro é correcto, devólvese o token *JWT* que se almacena mediante a librería CrossSecureStorage [15] (obtida mediante NuGet), que consiste nun diccionario clave-valor que se almacena como datos encriptados da aplicación.

US02 - Autenticación cliente

Novamente, xeramos un JWT ca estrutura amosada na figura 7.3 para o inicio de sesión como cliente. Neste caso sen embargo, nos claims, non estaría o campo *Commerce*.

No *frontend*, implementamos unha *TabbedPage* cun formulario de login na primeira *ContentPage*. Unha *TabbedPage* consiste nunha vista con pestañas que o usuario pode deslizar a dereita e esquerda para navegar polas distintas páxinas. Neste caso temos a de login inicialmente, ca posibilidade de facer un deslizamento á dereita e situarse no formulario de rexistro.

Centrándonos na de login, o usuario terá que introducir usuario e contraseña, o que xerará a petición POST para o *backend*, que validará eses datos e devolverá o token JWT no caso de ser correctos.

Se a autenticación é correcta, creamos, como na aplicación *TrocosComercios*, unha *NavigationPage* que contén o menú principal da aplicación.

US04 - Pechar sesión como cliente

Como utilizamos os tokens JWT, pechar a sesión so afecta ao *frontend*, co cal so temos que engadir no menú o botón de *Cerrar sesión*, e cando o usuario pulse nel, eliminamos o token do CrossSecureStorage e rediriximos á pantalla de autenticación/rexistro.

US03 - Autenticación comerciante

En canto ao *frontend*, temos unha *ContentPage* cun formulario de login, que cando o usuario completa, xera unha petición POST contra o *backend*.

Se o login é correcto, o backend devolve o token *JWT*, que contén nos claims o nome de usuario e o comercio que ten asociado, e créase unha nova *Navigation Page* que contén a páxina principal da aplicación co menú. A *NavigationPage* de Xamarin implementa a interfaz *INavigation*, que ofrece unha serie de métodos para navegar entre páxinas, que despois utiliza a navegación nativa da plataforma onde se instale a aplicación.

Na figura 7.3 podemos observar a estrutura de dito token, e como se firma.

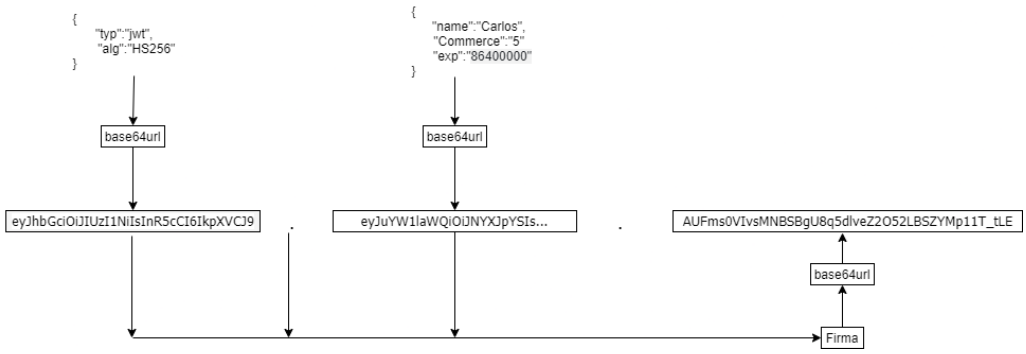

Figura 7.3: Formato token JWT

US05 - Pechar sesión como comerciante

No caso da aplicación de TrocosComercios faise o mesmo que na historia anterior, simplemente engadimos o botón de *Cerrar sesión* ao menú e rediriximos á pantalla de login.

7.5 Sprint 3: Búsqueda e visualización de comercios

7.5.1 Análise

Unha vez temos lista a xestión de usuarios e temos un mantemento de sesión en ambas aplicacións móbiles, podemos pasar á seguinte parte do proxecto, que é a búsqueda e visualización de comercios. A continuación descompoñemos as historias de usuario en tarefas, nas táboas 7.6, 7.7, 7.8, 7.9, 7.10 e 7.11:

US07 - Visualizar comercio
Como comerciante autenticado quero ver os datos do meu comercio.
Tarefas
Creación dun endpoint no *backend* que devolva un comercio según quen é o cliente que o posee.
Creación da vista en Xamarin que os datos do comercio.

Táboa 7.6: Sprint 2: Visualizar comercio

US11 - Buscar comercio
Como cliente autenticado quero buscar un comercio buscando por distintos parámetros.
Tarefas
Creación dun endpoint no *backend* que devolva unha lista de comercios filtrados pola cadea de texto introducida polo usuario.
Creación da vista en Xamarin que conteña a entrada de texto e devolva a lista de resultados.

Táboa 7.7: Sprint 2: Buscar comercio

US13 - Ver comercios
Como cliente autenticado quero visualizar os comercios que sigo.
Tarefas
Creación dun endpoint no *backend* que devolva os comercios aos que segue un cliente.
Creación da vista en Xamarin que conteña a lista de comercios aos que segue o cliente autenticado.

Táboa 7.8: Sprint 2: Ver comercios

US14 - Ver detalles de comercio
Como cliente autenticado quero visualizar os detalles dun comercio.
Tarefas
Creación dun endpoint no *backend* que devolva os datos dun comercio.
Creación no *frontend* dunha vista que conteña os detalles dun comercio.
Engadir a navegación a detalles dende a búsqueda de comercios.
Engadir a navegación a detalles dende a vista de comercios aos que segue un cliente.

Táboa 7.9: Sprint 2: Ver detalles de comercio

US15 - Seguir comercio
Como cliente autenticado quero seguir a un comercio.
Tarefas
Creación dun endpoint no *backend* que reciba un usuario e un comercio e cre a relación.
Validar no *backend* que a relación a crear non exista previamente.
Engadir un botón no *frontend* para seguir na vista de búsqueda de comercios de seguir comercio se o cliente autenticado non o segue.
Engadir un botón no *frontend* para seguir na vista de detalle de comercio se o cliente autenticado non o segue.

Táboa 7.10: Sprint 2: Seguir comercio

US16 - Deixar de seguir comercio
Como cliente autenticado quero deixar de seguir a un comercio.
Tarefas
Creación dun endpoint no *backend* que reciba un usuario e un comercio e elimine a relación.
Validar no *backend* que a relación a eliminar exista previamente.
Engadir un botón no *frontend* para deixar de seguir na vista de búsqueda de comercios de seguir comercio se o cliente autenticado xa o segue.
Engadir un botón no *frontend* para deixar de seguir na vista de detalle de comercio se o cliente autenticado xa o segue.
Engadir un botón no *frontend* para deixar de seguir na vista de comercios aos que o cliente segue.

Táboa 7.11: Sprint 2: Deixar de seguir comercio

7.5.2 Deseño e implementación

US07 - Visualizar comercio

Para visualizar o comercio, engadimos un botón no menú lateral *Ver datos de comercio* que se o comerciante pulsa, a aplicación amosa unha *Content Page* que contén a información do seu comercio.

US11 - Buscar comercio

Para a búsqueda, barallouse a posibilidade de crear filtros avanzados, pero optouse por ter unha única entrada na que o usuario poda introducir unha cadea de texto que filtrará os

comercios que teñan dita cadea de texto nalgún dos seus campos. Unha vez pulsa buscar, os resultados amosaranse en forma de lista debaixo do filtro de búsqueda.

No backend esta búsqueda realízase utilizando os lambdas de LinQ [16], que permiten filtrar fácilmente as entidades recuperadas da base de datos. Utilízase *Include* para realizar o JOIN necesario coa entidade *CommerceType* para incluír a información do tipo de comercio.

```
// GET: api/Commerces/search?Query=15005
[HttpGet("search")]
public ActionResult<IEnumerable<Commerce>> GetByQuery([FromQuery]
    string query)
{
    var commerce = _context.Commerce.Include(c => c.OType).Where(c
    => c.Address.Contains(query) || c.Cp.Contains(query)
    || c.Name.Contains(query) || c.Province.Contains(query) ||
    c.OType.Name.Contains(query) ||
    c.State.Contains(query) || c.Location.Contains(query)
    ).ToList();

    return Ok(commerce);
}
```

En cada elemento da lista amosamos a información máis importante de cada comercio para que o cliente obteña a información máis importante sen ter que ir ao detalle do comercio. Esta información consiste no nome do comercio, a súa dirección, e o icono que representa o tipo de comercio que é.

US13 - Ver comercios

Para ver os comercios aos que o cliente autenticado segue, engádese un botón de *Mis comercios* no menú lateral, que amosa unha *Content Page* cun *List View* como mesmo formato que o resultado das búsquedas de comercios, para manter a consistencia dentro da aplicación.

US14 - Ver detalles de comercio

Unha vez temos as funcionalidades de búsqueda e ver comercios aos que un cliente segue, podemos pasar a implementar a vista cos detalles dun comercio, na que se amosará información ampliada do comercio e as súas ofertas, divididas en dúas listas, as dispoñibles para o cliente e as bloqueadas. A esta vista accederase dende as vistas de *Búsqueda* e *Mis comercios*, pulsando no comercio en cuestión do que se queren ver os detalles, de forma que se navegará a él coa operación *Push Async* da interfaz *INavigation*, que engadirá a páxina á pila de navegación, puidendo voltar atrás dende o menú contextual.

US15 - Seguir comercio

Posteriormente a ter tódalas vistas dende onde se vai invocar, pasamos a engadir os botóns de seguir comercio ás vistas de detalle de comercio e búsqueda de comercio. Consistirá nun botón que so aparecerá se o cliente non segue xa a dito comercio, polo que temos que engadir unha condición a cada elemento da lista para comprobalo.

US16 - Deixar de seguir comercio

Por outro lado, temos a funcionalidade contraria á anterior, que é deixar de seguir un comercio. Estará nos mesmos lugares que seguir comercio, pero nos casos nos que o cliente xa siga ao comercio en cuestión. Adicionalmente estará tamén na vista dos comercios aos que segue o usuario.

7.6 Sprint 4: Xestión de ofertas

7.6.1 Análise

Nesta iteración pásase a implementar a toda a funcionalidade arredor das ofertas, posto que no sprint anterior se abordou todo o relativo aos comercios, que son os que conterán ditas ofertas. A continuación desgránase a descomposición en tarefas nas táboas 7.12, 7.13 e 7.14:

US06 - Alta de oferta
Como comerciante autenticado quero dar de alta unha oferta.
Tarefas
Creación dun endpoint no *backend* dunha operación POST que reciba unha oferta e a de de alta.
Validacións no cliente de que os campos introducidos son correctos.
Creación da vista que conteña un formulario no que o usuario poda introducir os campos necesarios.

Táboa 7.12: Sprint 3: Alta de oferta

US17 - Ver ofertas
Como cliente autenticado quero visualizar as ofertas dispoñibles para min.
Tarefas
Creación dun endpoint no *backend* que devolva a lista de ofertas dispoñibles para o cliente que se envíe por parámetro.
Creación da vista en Xamarin que conteña a lista de ofertas dispoñibles para o usuario de tódolos comercios aos que segue.

Táboa 7.13: Sprint 3: Ver ofertas

US18 - Ver detalle de oferta
Como cliente autenticado quero visualizar os detalles dunha oferta dispoñible para min.
Tarefas
Creación dun endpoint no *backend* que devolva toda a información dunha oferta.
Creación da vista en Xamarin que conteña a información dunha oferta.
Engadir posibilidade de navegar aos detalles dunha oferta dende os detalles dun comercio.
Engadir posibilidade de navegar aos detalles dunha oferta dende as ofertas dispoñibles para un cliente.

Táboa 7.14: Sprint 3: Ver detalle de oferta

7.6.2 Deseño e implementación

US06 - Alta de oferta

A alta de oferta realizarase mediante unha petición *POST* ao backend, cuxa información se recollerá dende un formulario que o comerciante terá que completar dende a aplicación TrocosComercios.

A información que introduza será validada pola propia aplicación, amosando unha mensaxe de erro no caso de que algún dos campos sexa erróneo. Unha vez validada, realizarase a petición ao backend, que devolverá un *true* en caso de éxito.

Máis adiante engadirase a funcionalidade de enviar unha notificación a tódolos clientes que sexan seguidores do comercio que realiza a alta.

US17 - Ver ofertas

Para a visualización de ofertas, engádese á pantalla de inicio de *Trocos* un *ListView* que conterá as ofertas que devolva o *backend* cando lle enviemos o identificador do cliente autenticado. Na figura 7.4 podemos ver a ventá.

Figura 7.4: Páxina de visualización de ofertas

US18 - Ver detalle de oferta

Como xa fixemos cos comercios na aplicación de *Trocos*, para as ofertas tamén se engadirá a funcionalidade de visualizar os seus detalles. Para elo, permitirase pulsar sobre unha oferta, xa sexa dende a vista do listado de ofertas dispoñibles ou dende os detalles dun comercio, onde se amosan as ofertas dispoñibles para o cliente de ese comercio en concreto.

7.7 Sprint 5: Implementación de QRs, consumición de ofertas e estadísticas

Unha vez temos lista a xestión de ofertas, neste *sprint* tratamos a consumición de ofertas e as estadísticas dos comercios.

En canto a como consumir ofertas, realízase mediante o escaneo de códigos QR que conteñen un token *JWT* coa información relativa á oferta e ao cliente que desexa consumila.

Tamén incluímos a funcionalidade para os comerciantes de poder visualizar estadísticas sobre o seu comercio e as ofertas que vai dando de alta.

Amosamos a descomposición en tarefas nas táboas 7.15, 7.16 e 7.17:

7.7.1 Análise

US19 - Ver QR de oferta
Como cliente autenticado quero visualizar o código QR dunha oferta para poder consumila.
Tarefas
Creación dun endpoint no *backend* dunha operación que devolva un token JWT ca información a ser representada polo QR.
Crear no *frontend* unha imaxe que conteña o código QR co token proporcionado polo *backend*

Táboa 7.15: Sprint 4: QR de oferta

US09 - Consultar estadísticas
Como comerciante autenticado quero consultar estadísticas do meu comercio.
Tarefas
Creación dun endpoint no *backend* que devolva as estadísticas a amosar para un comercio.
Creación da vista en Xamarin que conteña a información das estadísticas co comercio do comerciante autenticado.

Táboa 7.16: Sprint 4: Consultar estadísticas

US08 - Escanear oferta
Como comerciante autenticado quero escanear un código QR dun cliente.
Tarefas
Obter o token *JWT* a partir ca imaxe que contén o código QR
Creación dun endpoint no *backend* que reciba o token *JWT* coa oferta e o cliente e marque como consumida esa oferta para ese cliente.
Validar que a oferta sexa válida.
Validar que esa oferta non fose previamente consumida polo cliente.
Validar que a oferta pertence ao comercio que está facendo a petición.

Táboa 7.17: Sprint 4: Escanear oferta

7.7.2 Deseño e implementación

US19 - Ver QR de oferta

Para evitar posibles falsificacións de códigos QR, decidiuse utilizar un token JWT como o da figura 7.3, para transmitir a información que conterá o código, de forma que un código QR so poida ser creado polo backend da aplicación. Por este motivo, créase unha operación *GET* no *backend* que devolve un token *JWT* co nome de login do cliente que desexa consumir a oferta e o identificador da oferta nos seus claims. Este token reenvíase ao backend posteriormente cando un comerciante escanea o código QR ca aplicación de *TrocosComercios*.

Tomouse a decisión de recurrir aos token *JWT* nesta operación para evitar códigos QR fraudulentos, xa que se un usuario da plataforma averiguase o identificador de outro usuario, podería xerar un código QR co nome do outro usuario, e consumir unha oferta que non lle correspondería. Esto evítase mediante os token, xa que estes se atopan firmados, desta forma o backend pode contrastar a validez do token antes de consumir a oferta.

Cando se implementen as notificacións, este método enviará unha notificación ao cliente que consume a oferta indicándolle o éxito da operación.

Para crear unha imaxe que conteña o QR, foi necesario crear un servizo *QRService* que implementase o método *ConvertImageStream* dentro da solución de cada unha das plataformas, posto que debe facerse de xeito distinto en cada unha. Por tanto, créase unha interfaz *IQRService* na solución principal, e despois un servizo en cada unha das solucións de cada plataforma implementando dito método.

Además, inclúese esta interfaz no *DependencyService*. Este encargarse posteriormente de implementar o método según a plataforma na que se execute a aplicación. Na figura 7.5 podemos ver un diagrama do seu funcionamento

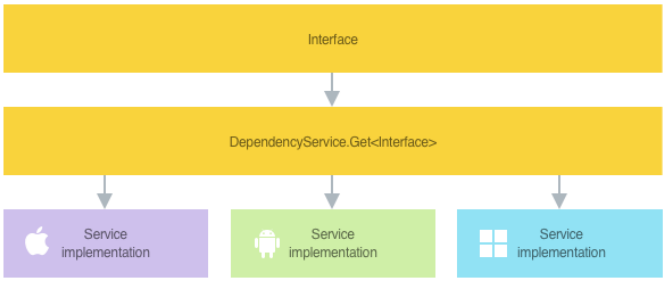

Figura 7.5: Funcionamento de Dependency Service

No frontend créase unha *ContentPage* que contén a información da oferta, así como a imaxe do código QR, que ten a posibilidade de ser ampliada se o cliente pulsa nela, para facilitar o seu escaneado. Na información amosada, inclúese a data de caducidade da oferta, a súa descrición, e o nome e o icono do comercio que a emite, así como a categoría á que pertence (que tamén determina a cor do recadro e do nome do comercio).

Na figura 7.6 podemos observar a ventá.

Figura 7.6: Páxina de detalle de oferta

US09 - Consultar estadísticas

Para a consulta de estadísticas, engadimos ao menú de *TrocosComercios* a opción *Estadísticas*. Se un comerciante pulsa nela, accederá a unha *ContentPage* onde poderá visualizar o número de veces que foi consumida cada unha das súas ofertas, así como cantas ofertas foron consumidas de cada categoría.

US08 - Escanear oferta

Para esta funcionalidade, créase unha petición *POST* no *backend* que recibirá o token ca información do cliente e da oferta a ser consumida, e extraerá do token de autorización o identificador do comercio do comerciante que envía a petición, para validar que a oferta pertenza ao seu comercio (no caso contrario devolverá un erro).

No frontend, engádese no menú da aplicación *TrocosComercios*, a opción de *Escanear*, e tamén se convirte na páxina principal para facilitar esta operación aos comerciantes, de xeito que sexa a primeira opción que teñen cando se autentican, para axilizar o proceso.

A vista consistirá nun botón co texto *Pulse aquí para empezar a escanear*, que abrirá unha nova *ZXingScannerPage*, que se obtén da librería *ZXing ZXing*, instalada mediante NuGet. Esto abrirá a cámara do dispositivo móbil e chamará ao método que lle especifiquemos cando detecte un código QR.

Cando o faga, o método que implementamos chamará ao backend, enviándolle o token do código QR en cuestión. O backend informaranos de se a operación foi correcta, e amosaremos unha mensaxe de confirmación ao comerciante.

Se a petición é válida, engadiremos un novo *CustomerOffer* (a táboa da relación *ManyToMany* entre clientes e ofertas) para dar por consumida a oferta para dito cliente.

7.8 Sprint 6: Xestión de notificacións

7.8.1 Análise

Este *sprint* aborda a xestión de notificacións que se vai a utilizar nas aplicacións móbiles. Como mencionamos anteriormente, optouse por utilizar Azure Notifications Hub, que ademáis de otorgar moita escalabilidade e a posibilidade de envíos masivos de notificacións, tamén incorpora un flexible sistema de envío de notificacións a usuarios concretos, mediante o uso de etiquetas.

Estas etiquetas poden ser calquer cadea de ata 120 caracteres, permitindo caracteres alfanuméricos e unha serie de non alfanuméricos. Cando se rexistra un dispositivo, faise cunha lista de etiquetas. Deste xeito, cando se quere enviar unha notificación, se se especifica algunha etiqueta ou etiquetas, esta notificación so se enviará aos dispositivos rexistrados con

dita etiqueta.

Neste proxecto, utilizamos as etiquetas para dúas funcionalidades. A primeira, para cada cliente, engadimos unha etiqueta por cada comercio ao que segue, desta forma podemos enviar notificacións cuxa etiqueta sexa o comercio que emite unha nova oferta, recibindo a notificación os seus seguidores. Por outra parte, gardamos tamén unha etiqueta co nome do usuario, para que cando consuma unha oferta, podamos enviarlle unha notificación directamente a ese usuario, para que reciba confirmación da consumición da oferta.

Concretamente, engádense as notificacións para os US06 - Alta de oferta e US08 - Escanear oferta. Tamén é necesario engadir unha modificación aos US15 - Seguir comercio e US16 - Deixar de seguir comercio, xa que é preciso actualizar a lista de etiquetas dun cliente cando segue ou deixa de seguir a un comercio; e os US02 - Autenticación cliente e US04 - Pechar sesión como cliente, porque cando un usuario inicia sesión debe ser rexistrado no hub de notificacións, e cando pecha sesión, debe eliminarse para deixar de recibilas. No caso da alta, notificarase a tódolos clientes que sexan seguidores do comercio que a crea; e no caso de escanear oferta, notificarase ao cliente que consume a oferta do éxito da operación. Na figura 7.7 podemos ver un diagrama no que se observa este fluxo.

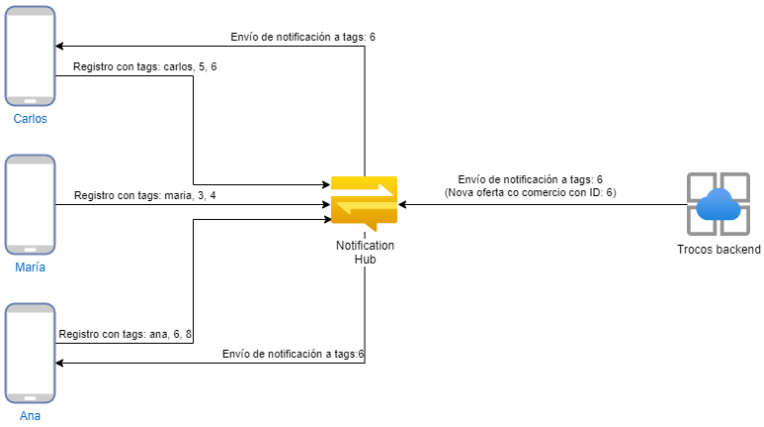

Figura 7.7: Diagrama de fluxo de etiquetas

7.8.2 Deseño e implementación

US06 - Alta de oferta

Cando unha oferta é dada de alta con éxito, utilizamos o sistema de etiquetas de *Azure Notifications Hub*, para notificar a tódolos clientes que estén rexistrados no Hub de notificacións coa etiqueta co identificador do comercio que emite dita oferta.

US08 - Escanear oferta

No caso de que a oferta escaneada sexa correcta e se sumen os puntos ao usuario, utilizamos o seu login para enviar unha notifación aos dispositivos que conteñan dita etiqueta, de forma que o usuario recibirá unha notificación informándolle do éxito da operación.

US15 - Seguir comercio

Cando un cliente segue a un comercio, realízase un refresco do seu rexistro no Hub de notificacións, de forma que se engade como nova etiqueta o identificador do comercio que acaba de seguir, para recibir notificacións de dito comercio no caso de que emita novas ofertas.

US16 - Deixar de seguir comercio

Se un cliente decide deixar de seguir a un comercio, debe realizarse tamén unha actualización do seu rexistro no Hub de notificacións, eliminando da lista de etiquetas o comercio ao que deixa de seguir, de forma que deixe de recibir avisos no caso de que de de alta novas ofertas.

US02 - Autenticación cliente

Cando un cliente inicia sesión nun dispositivo, realízase un rexistro no *Azure Notifications Hub* co identificador do dispositivo actual, xunto ca lista de etiquetas que contén o nome de usuario e os identificadores de tódolos comercios aos que segue. Desta forma, o dispositivo no que acaba de iniciar sesión queda rexistrado para recibir notificacións, tanto dos comercios aos que segue para cando emitan ofertas, como para cando consuma unha oferta.

US04 - Pechar sesión como cliente

No caso de que un cliente peche sesión, realízase a eliminación do rexistro no Hub de notificacións, para que deixe de recibilas nese dispositivo.

7.9 Sprint 7: Conexión ca API de Google e mapa

7.9.1 Análise

Neste *sprint* trátase de implementar a funcionalidade de visualizar o mapa do mundo no que se vexan pines nas localizacións dos distintos comercios dados de alta no sistema. Amosamos a descomposición en tarefas da historia de usuario na táboa 7.18:

US12 - Ver mapa
Como cliente autenticado quero visualizar os comercios no mapa.
Tarefas
Crear un endpoint no backend que nos devolva as coordenadas dos comercios.
Crear unha conta na API de Google para conseguir un código de conexión para a API de *GoogleMaps*.
Configurar a aplicación de *Trocos* para poder visualizar mapas de *GoogleMaps*
Crear unha ventana que conteña o mapa e permita visualizalo, con pines coas localizacións dos comercios.
Solicitar os permisos de localización ao usuario.

Táboa 7.18: Sprint 6: Ver mapa

7.9.2 Deseño e implementación

US12 - Ver mapa

Para incorporar o mapa no *frontend*, incorporamos ao proxecto de *Trocos* a librería de *Xamarin.Forms.GoogleMaps* [17] para poder utilizar o mapa de *GoogleMaps*. Debemos incorporar no MainActivity (na solución de Android) a incialización do compoñente.

Creamos un perfil para a aplicación de Trocos na api de Google Cloud Platform [18], e habilitamos a api de *Google Maps* para obter unha clase de API. Dita clave debe ser introducida no manifesto de Android da seguinte forma:

```
<meta-data android:name='com.google.android.maps.v2.API_KEY'
    android:value='CLAVEAPI'/>
```

Tamén é necesario engadir ao manifesto os seguintes permisos:

- AccessCoarseLocation
- AccessFineLocation
- Internet

Posteriormente, engadimos ao menú unha nova entrada chamada *Mapa*, que leva a unha nova *ContentPage* na que se inclúe o mapa con pines representando as distintas localizacións dos comercios. Seleccionar un deses pines nos levará aos detalles de dito comercio.

Capítulo 8

Conclusións e traballo futuro

8.1 Conclusións

Unha vez finalizado o proxecto, danse por acadados os obxetivos inciais do mesmo.

Por un lado buscábase unha aplicación móbil para dotar ao comercio local dunha ferramenta que os acerque a un comercio electrónico que moitos non se poden permitir; e por outro lado crear un portal ao alcance da man para tódolos clientes que queiran descubrir e agrupar os comercios que os rodean.

Por outra parte, o sistema de códigos QR axiliza este intercambio entre cliente e comerciante, facéndoo rápido e pouco engorroso, que era un dos riscos a evitar. Tamén o sistema de categorías convida ao cliente a utilizar a aplicación para acceder a mellores ofertas, o que para os comerciantes tamén será beneficioso porque, además de atraer clientes, poderán ver qué ofertas funcionan mellor e obter información do tipo de clientela que teñen.

Para rematar quérese facer énfase nos coñecementos adquiridos polo alumno, que se ben, por motivos de traballo, xa traballara previamente con *Xamarin*, neste proxecto leváronse a cabo funcións máis avanzadas que lle permitiron aprender novas técnicas e recursos, como un mellor uso do patrón *MVVM* e a potencia dos servizos de *Azure*, que proporcionan gran comodidade á hora de desplegar e gran capacidade e potencia.

Tamén se aprendeu moito sobre *LinQ* e a forma de traballar cas entidades que esta librería ofrece, que ben utilizada otorga unhas ferramentas moi interesantes para realizar consultas á base de datos dunha forma moi programática, acercándose moito, co uso de lambdas, ao tratamento de listas de *Java* e outras linguaxes, facilitando moito este proceso.

Outro coñecemento adquirido foi o tratamento de notificacións mediante *Azure Notifications Hub*, que otorga gran versatilidade co sistema de etiquetas e uns niveis de escalabilidade máis que suficientes para este proxecto, cunha configuración relativamente sinxela podemos obter un sistema de notificacións moderno e flexible.

8.2 Traballo futuro

Este proxecto senta unha base inicial para o desenvolvemento dunha aplicación completa, con potentes e interesantes melloras para continuar unha liña de desenvolvemento que pode desembocar nunha plataforma moito máis completa. A continuación fálase das que o alumno considera máis importantes:

- **Permitir pagos a través da aplicación.** A través dos propios códigos QR, permitir ao usuario pagar a propia oferta que está consumindo. Desta forma poderase mellorar o sistema de puntos e categorías, e pode ser interesante para o modelo de negocio, quizais recibindo un porcentaxe dos pagos que se realicen a través da aplicación.
- **Creación dun portal web.** Crear un portal web que permita tanto a clientes, pero sobre todo comerciantes, conectarse á plataforma a través da web, e onde se poden incluír funcionalidades máis complexas como unhas estadísticas para os comerciantes máis avanzadas e incluso un envío de notificacións que non dependa so das altas de ofertas.
- **Sistema de valoracións.** Permitir aos clientes realizar valoracións tanto sobre os comercios como sobre as ofertas, de forma que os comercios reciban información do que gusta e non, e os clientes podan informarse a partir da opinión de outros clientes e das súas experiencias.
- **Publicar as aplicacións no resto de plataformas dispoñibles.** Como o proxecto foi realizado en Xamarin, as aplicacións móbiles poderían ser implementadas nas plataformas iOS e UWP, con algo de configuración adicional.

Apéndices

Relación de Acrónimos

API Application Programming Interface (Interfaz que define unha serie de operacións para que podan ser consumidas por outras). 21

SCM Software Configuration Management. 24

US User Story (Historia de usuario). 14

Glosario

backend Capa dunha aplicación que se encarga de tódolos procesos necesarios para que a aplicación funcione de forma correcta. Parte ”invisible”. 21

big data Conxunto de datos cuxo volume é tan grande que as ferramentas tradicionais de procesamento de datos son totalmente ineficaces, polo que surxe a necesidade de aplicacións informáticas para o su tratamento. 3

control de versións Xestión dos diversos cambios que se realizan sobre os elementos de algún producto ou a configuración do mesmo. 24

frontend Capa visible dunha aplicación, ca que usuario interactúa. Interfaz de usuario. 21

gamificar Aplicación de elementos propios de xogos a outros ámbitos, ca fin de potenciar a motivación do seu uso. 3

mockup Esquema ou bosquexo da interfaz de usuario dun sistema. 12

Bibliografía

[1] "Scrum guides." [En liña]. Dispoñible en: https://www.scrumguides.org/

[2] "Páxina web de redmine." [En liña]. Dispoñible en: https://www.redmine.org/

[3] "Documentación de .net." [En liña]. Dispoñible en: https://docs.microsoft.com/es-es/dotnet/

[4] "Páxina de descarga de sql server." [En liña]. Dispoñible en: https://www.microsoft.com/es-es/sql-server/sql-server-downloads

[5] "Qué es azure." [En liña]. Dispoñible en: https://azure.microsoft.com/es-es/overview/what-is-azure/

[6] "Documentación de c#." [En liña]. Dispoñible en: https://docs.microsoft.com/es-es/dotnet/csharp/

[7] "Qué es xamarin." [En liña]. Dispoñible en: https://docs.microsoft.com/es-es/xamarin/get-started/what-is-xamarin

[8] "Documentación de entity framework." [En liña]. Dispoñible en: https://docs.microsoft.com/es-es/ef/

[9] "Páxina web de visual studio." [En liña]. Dispoñible en: https://visualstudio.microsoft.com/es/

[10] "Páxina web de azure notifications hub." [En liña]. Dispoñible en: https://azure.microsoft.com/es-es/services/notification-hubs/

[11] "Páxina web de nuget." [En liña]. Dispoñible en: https://www.nuget.org/

[12] "Páxina web de firebase cloud messaging." [En liña]. Dispoñible en: https://firebase.google.com/docs/cloud-messaging

[13] "Páxina web de git." [En liña]. Dispoñible en: https://git-scm.com/

[14] "Portal web de github." [En liña]. Dispoñible en: https://github.com/

[15] "Repositorio do plugin crosssecurestorage." [En liña]. Dispoñible en: https://github.com/sameerkapps/SecureStorage/blob/master/SecureStorage/Plugin.SecureStorage/CrossSecureStorage.cs

[16] "Documentación de linq." [En liña]. Dispoñible en: https://docs.microsoft.com/es-es/dotnet/csharp/programming-guide/concepts/linq/introduction-to-linq-queries

[17] "Repositorio do plugin googlemaps." [En liña]. Dispoñible en: https://github.com/amay077/Xamarin.Forms.GoogleMaps

[18] "Páxina web de google cloud platform." [En liña]. Dispoñible en: https://cloud.google.com/