

UNIVERSIDAD DE SONORA

DIVISIÓN DE INGENIERÍA

Departamento de Ingeniería Industrial

Desarrollo y Administración de el sistema “Botón de Proximidad”

Reporte de Prácticas Profesionales

PRESENTA:

Mario Alberto Lugo Talamante

INGENIERO EN SISTEMAS DE INFORMACIÓN

Gerardo Sánchez Schmitz

ÍNDICE GENERAL

ÍNDICE GENERAL	2
ÍNDICE DE FIGURAS	3
1. INTRODUCCIÓN	4
2. DESCRIPCIÓN DEL CONTEXTO	6
3 FUNDAMENTO TEÓRICO DE LAS HERRAMIENTAS Y CONOCIMIENTOS APLICADOS	8
4 DESCRIPCIÓN DETALLADA DE LAS ACTIVIDADES REALIZADAS	16
5 ANÁLISIS DE LA EXPERIENCIA ADQUIRIDA	20
6 CONCLUSIONES Y RECOMENDACIONES	22
7 REFERENCIAS BIBLIOGRÁFICAS Y VIRTUALES	23
ANEXOS	24

ÍNDICE DE FIGURAS

2.1 LOGO DE LA EMPRESA	6
3.1 APLICACIÓN DE POLICÍA	8
3.2 BOTÓN DE ALERTA	9
3.3 LISTA DE VECINOS	9
3.4 CHAT DEL COMITÉ	10
3.5 ALERTA A ATENDER	10
3.6 VISTA POLICÍA EN CAMINO	10
3.7 CALIFICACIÓN POLICIA A USUARIO	10
3.8 CALIFICACIÓN USUARIO A POLICÍA	10
3.9 PANEL DE ADMINISTRACIÓN	15

1. INTRODUCCIÓN

En la reglamentación de Universidad de Sonora se tiene contemplado que en todos los planes de estudio se incluyan actividades de vinculación con el sector social o productivo con el propósito de complementar la formación de los estudiantes a través de la aplicación de los conocimientos obtenidos en las diversas materias de las carreras. Así, la carrera de Ingeniería en Sistemas de Información, del Departamento de Ingeniería Industrial, incluye en su plan de estudios las prácticas profesionales con valor 20 créditos, que son equivalentes a 340 horas.

Para cumplir con ese requisito, pedí de manera formal a la empresa donde actualmente trabajo llamada QuantumBit S.A. de C.V., que me aceptara hacer las prácticas ahí mientras laboro. En tal empresa tengo aproximadamente un año trabajando, empecé en el 2015 en el puesto de Front-End Developer, a los 4 meses se me asignó a Full-Stack Developer, desarrollando aplicaciones móviles híbridas. A los 10 meses, me nombraron Project Manager.

Éste proyecto se desarrolló de Julio a Septiembre del 2016, con los propósitos de cumplir con ese requisito curricular, de complementar la formación recibida en las clases teóricas y prácticas de la carrera y adquirir la práctica necesaria para desarrollar un trabajo similar.

El presente trabajo es una memoria de las actividades realizadas durante el tiempo en que se desarrollan las prácticas profesionales en la empresa QuantumBit. El contenido restante del documento tiene los siguientes apartados: la descripción del contexto, en el cual se explica la operación y características de la unidad receptora, en éste caso QuantumBit. En la siguiente se pone el fundamento teórico de las herramientas y conocimientos aplicados durante el desarrollo de las prácticas profesionales, en particular los relacionados con el desarrollo de aplicaciones móviles, la administración de proyectos y bases de datos. Posteriormente se presenta una descripción detallada de las actividades realizadas, divididas en períodos según los reportes parciales que se hicieron. Enseguida se expone una valoración de varios temas relacionados la experiencia adquirida en ese período. Por último están las conclusiones y recomendaciones que resultaron de la experiencia que se tuvo en esa entidad receptora.

1.1 BREVE EXPLICACIÓN DEL PROYECTO O ACTIVIDAD

El proyecto “Botón de Proximidad” es una propuesta de seguridad pública, del gobierno municipal, que su fin es reducir el tiempo de respuesta de los policías para emergencias de los ciudadanos. El proyecto consiste en una aplicación llamada “Hermosillo Seguro”, donde la bajan los miembros de comités, posteriormente se registran y su cuenta queda en espera de validación por el encargado de la administración del sistema. La otra aplicación es la del policía donde espera las alertas de emergencia que activan los miembros del comité. También está un panel de administrador donde vienen estadísticas y lo necesario para administrar éste proyecto.

1.2 OBJETIVOS

El objetivo general de desarrollar las prácticas en QuantumBit fue tratar de aplicar los conocimientos adquiridos en la carrera de Ingeniería en Sistemas de Información de la universidad de Sonora, complementar la formación obtenida durante la escuela y el tiempo de experiencia trabajando para posteriormente estar en mejor posibilidad de ejercer profesionalmente en estar mejor posibilidad de crecer profesionalmente.

Entre los principales objetivos específicos de realizar las prácticas fue lograr tener la experiencia para desarrollar y dirigir proyectos de aplicaciones móviles.

2. DESCRIPCIÓN DEL CONTEXTO

QuantumBit es una empresa de consultoría y desarrollo de software, actualmente lleva aproximadamente 2 años operando. Se desarrollan sistemas web y aplicaciones móviles para el sector gubernamental. Es una empresa con 12 colaboradores y un director. Se desarrolla en diferentes lenguajes como JavaScript, Ruby, Php, HTML y CSS. Y trabajamos bajo metodologías ágiles como SCRUM.

2.1 EQUIPAMIENTO E INSTALACIONES



Figura 2.1 Logo de la empresa

QuantumBit se localiza en Colosio #294, en el tercer piso; dentro de ese piso hay una oficina aproximadamente para capacidad para unas 15 personas, donde cada quien cuenta con su propio monitor y laptop Apple Macbook Pro de 15 pulgadas. Se cuenta con una conexión a internet de 200mb de velocidad. No se cuenta con servidores físicos, puros servidores virtuales del proveedor DigitalOcean.

2.2 ENTORNO DONDE SE UBICA LA UNIDAD RECEPTORA

La empresa solo se dedica al desarrollo de software para el gobierno municipal; QuantumBit es un micro negocio y se encuentra dentro de un grupo de micro negocios. En la oficina, hay otros 3 micro negocios, uno de renta de maquinaria, otro que es una constructora y otro que se encarga de la contabilidad de los micro negocios. Cada uno cuenta con menos de 12 empleados, ya que la filosofía que se tiene aquí de “empresa inteligente” no permite tener a más de 12 colaboradores en un micro negocio.

2.3 NORMATIVIDAD DE LA UNIDAD RECEPTORA

En cuanto a la normatividad, QuantumBit es una empresa que guía todo su quehacer por un conjunto de reglas, lineamientos, policial, manuales y procedimientos, los cuales son definidos por el director de la empresa.

En particular, cuenta con documentos de descripción de cada uno de los puestos de la empresa, también contiene todo el modelo de negocio de la empresa que incluye: misión, visión, diagrama de flujo, etc.

3 FUNDAMENTO TEÓRICO DE LAS HERRAMIENTAS Y CONOCIMIENTOS APLICADOS

3.1 APLICACIÓN DE USUARIO DEL COMITÉ Y POLICÍA

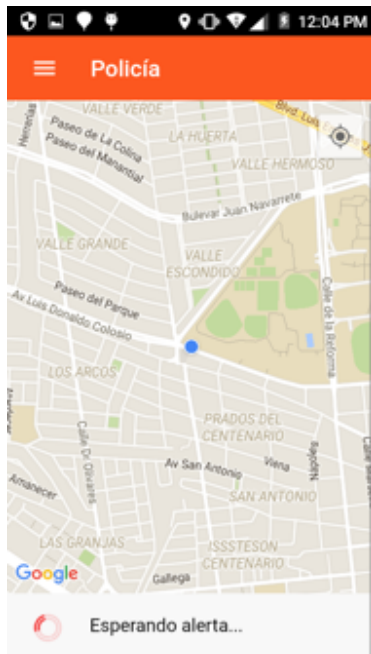


Figura 3.1. Aplicación Policías

La aplicación de usuario del comité y la del policía fueron hechas con Javascript, utilizando el framework de angular 2 (www.angular.io) y ionic 2 (<http://ionic.io/2>), que viene siendo un conjunto de herramientas que compilan aplicaciones híbridas. Me refiero a “aplicaciones híbridas” a las aplicaciones que son basadas y compiladas en base a un navegador web, Ionic se encarga de compilar la aplicación y agregarle funciones nativas, mediante plugins; por ejemplo, si se quiere utilizar la cámara, se necesita instalar un plugin nativo para la cámara mediante un comando por ejemplo: “ionic plugin add <NOMBRE-DEL-PLUGIN>”.

Las aplicaciones consumen los datos de una Api Restful que se explica en el apartado 3.3 API RESTFUL, y el tiempo real mediante socket.io.

El funcionamiento de ésta aplicación es básico, los policías tienen un dispositivo móvil con el sistema operativo Android; la aplicación de policía es un mapa, el cual se aprecia en la figura 3.1, donde está activa y mandando la ubicación en tiempo real al panel de administrador. La otra aplicación es la aplicación de los comités, que tiene un registro de usuarios, y al registrarse, la cuenta queda en espera de la validación. Ya que se valida, ya puede iniciar sesión y el usuario ya puede emitir alertas al oficial (figura 3.2), puede usar un chat de vecinos (figura 3.4) y puede ver la lista de vecinos (figura 3.3).



Figura 3.2. Botón de alerta

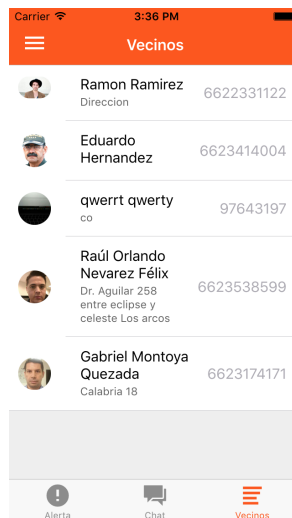


Figura 3.3. Lista de vecinos

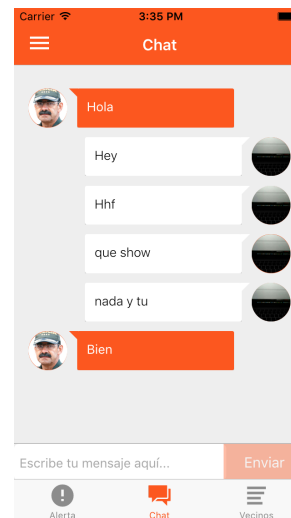


Figura 3.4. Chat de vecinos

Cuando el presidente activa la alarma, le llega la notificación al policía más cercano (figura 3.5), en caso de que el oficial no esté disponible o no acepte la alarma, se pasará al siguiente policía más cercano. Ya que el policía acepta, le manda una notificación a la persona que mandó la alerta (figura 3.6), y puede ver los datos del policía, puede llamarle, mándale mensaje, y tiene un cronómetro para ver el tiempo transcurrido (el presidente municipal prometió en un informe de gobierno que el tiempo sería de 8 minutos). Ya que el policía llegó a atender la alerta, puede calificar al usuario y poner el tipo de alerta (figura 3.7), y el usuario puede calificar al policías (figura 3.8). Si llega a ser una falsa alarma, el policía puede bloquear a los usuarios. Para esta aplicación se está capacitando a los miembros del comité, y a los policías ya se les capacitó.

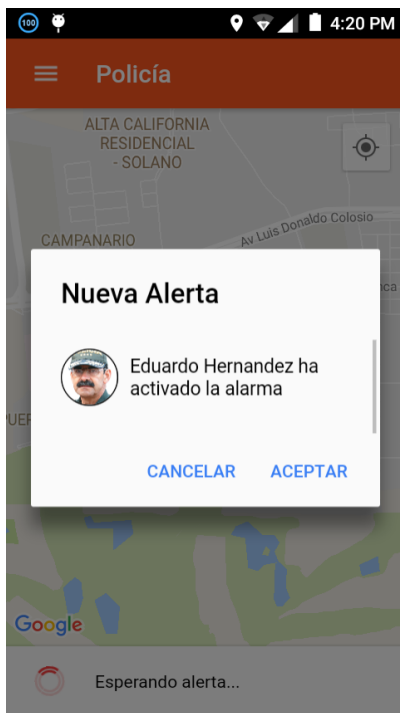


Figura 3.5. Alerta a atender

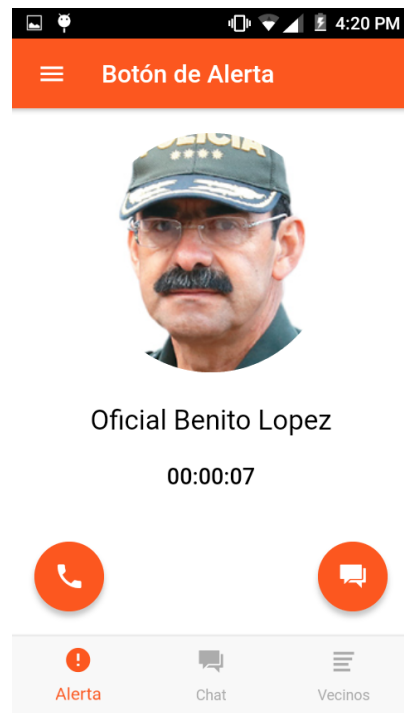


Figura 3.6. Vista de policía en camino

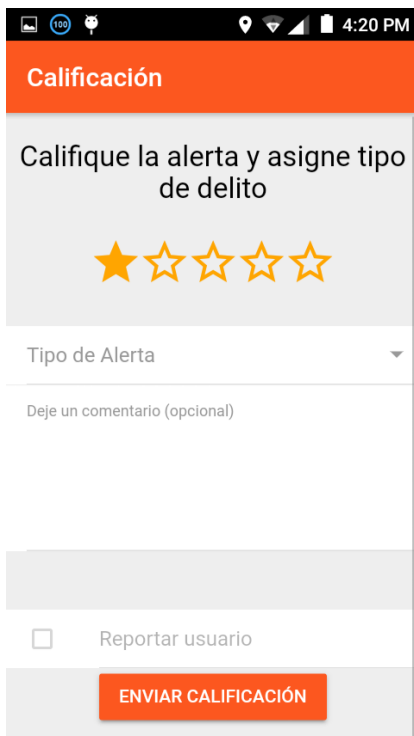


Figura 3.7. Calificación de usuario y tipo de alerta

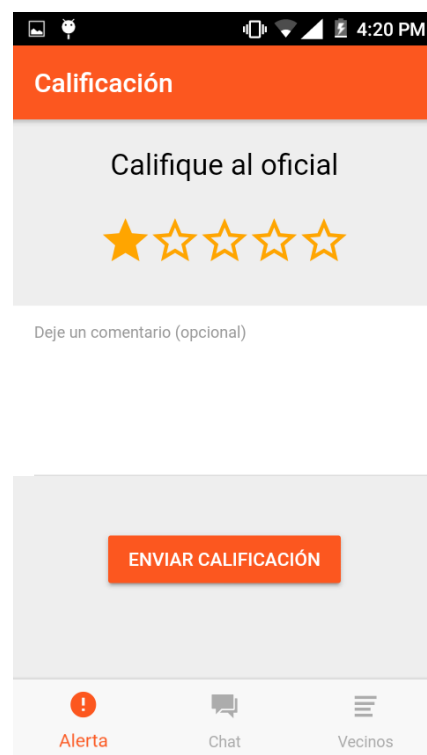


Figura 3.8. Calificación al Policía

3.2 SERVIDOR

Se adquirieron los servicios de un proveedor de servidores virtuales llamado Digital Ocean (www.digitalocean.com). Se está rentando un servidor virtual con Ubuntu 14.04. El servidor nos lo dieron sin nada instalado. Posteriormente se le instaló git para poder subir el código de los servicios. Después se le instaló Docker, que es un contenedor de aplicaciones, para no tener que instalarle dependencias al servidor.

Después se instaló MySQL para gestionar las bases de datos. El servidor se compone de una Api Restful hecha en Ruby on rails (apartado 3.3 Api Restful), unos servicios para el tiempo real hecha en NodeJs (apartado 3.4 Real Time), Mysql y el panel de administración hecho en angular 2 (apartado 3.5 Panel de administración).

Después de instalar docker, se creó un contenedor de ruby on rails y se ejecutó. Después se instaló y se montó NodeJs, RethinkDB y Redis, posteriormente se hablará de esto. A lo último se instaló el panel de administración.

3.3 API RESTFUL

La Transferencia de Estado Representacional (REST - Representational State Transfer) fue ganando amplia adopción en toda la web como una alternativa más simple a SOAP y a los servicios web basados en el Language de Descripción de Servicios Web (Web Services Description Language - WSDL). Ya varios grandes proveedores de Web 2.0 están migrando a esta tecnología, incluyendo a Yahoo, Google y Facebook, quienes marcaron como obsoletos a sus servicios SOAP y WSDL y pasaron a usar un modelo más fácil de usar, orientado a los recursos.

REST define un set de principios arquitectónicos por los cuales se diseñan servicios web haciendo foco en los recursos del sistema, incluyendo cómo se accede al estado de dichos recursos y cómo se transfieren por HTTP hacia clientes escritos en diversos lenguajes. REST emergió en los últimos años como el modelo predominante para el diseño de servicios. De hecho, REST logró un impacto tan grande en la web que prácticamente logró desplazar a SOAP y las interfaces basadas en WSDL por tener un estilo bastante más simple de usar.

Una de las características claves de los servicios web REST es el uso explícito de los métodos HTTP, siguiendo el protocolo definido por RFC 2616. Por ejemplo, HTTP GET se define como un método productor de datos, cuyo uso está pensado para que las aplicaciones cliente obtengan recursos, busquen datos de un servidor web, o ejecuten una consulta esperando que el servidor web la realice y devuelva un conjunto de recursos.

REST hace que los desarrolladores usen los métodos HTTP explícitamente de manera que resulte consistente con la definición del protocolo. Este principio de diseño básico establece una asociación uno-a-uno entre las operaciones de crear, leer, actualizar y borrar y los métodos HTTP. De acuerdo a esta asociación:

- se usa POST para crear un recurso en el servidor
- se usa GET para obtener un recurso
- se usa PUT para cambiar el estado de un recurso o actualizarlo
- se usa DELETE para eliminar un recurso
-

Una falla de diseño poco afortunada que tienen muchas APIs web es el uso de métodos HTTP para otros propósitos. Por ejemplo, la petición del URI en un pedido HTTP GET, en general identifica a un recurso específico. O el string de consulta en el URI incluye un conjunto de

parámetros que definen el criterio de búsqueda que usará el servidor para encontrar un conjunto de recursos. Al menos, así como el RFC HTTP/1.1 describe al GET.

La API REST se desarrolló en ruby on rails, Ruby on Rails, también conocido como RoR o Rails, es un framework de aplicaciones web de código abierto escrito en el lenguaje de programación Ruby, siguiendo el paradigma del patrón Modelo Vista Controlador (MVC). Trata de combinar la simplicidad con la posibilidad de desarrollar aplicaciones del mundo real escribiendo menos código que con otros frameworks y con un mínimo de configuración. El lenguaje de programación Ruby permite la metaprogramación, de la cual Rails hace uso, lo que resulta en una sintaxis que muchos de sus usuarios encuentran muy legible. Rails se distribuye a través de RubyGems, que es el formato oficial de paquete y canal de distribución de bibliotecas y aplicaciones Ruby.

La API se comunica a NodeJS mediante Redis, que es un servidor de caché de memoria que conoce la estructura de los datos que alberga, o dicho de otra forma, una caché de datos con estructura.

La idea de servidores de caché de memoria en los que se alberga información para su rápido acceso desde distintos puntos es muy antigua, y existen productos muy consolidados en el mercado, como el famoso Memcached, utilizado por sitios con muchos millones de usuarios como Wikipedia, Twitter, Youtube, WordPress, etc.

Se manejan todos los servicios web desde Ruby on Rails, y éste funciona directamente con MySQL.

3.4 REAL TIME (COMUNICACIÓN EN TIEMPO REAL)

La aplicación se hizo en NodeJS, que es un entorno Javascript del lado del servidor, basado en eventos. Node ejecuta javascript utilizando el motor V8, desarrollado por Google para uso de su navegador Chrome. Aprovechando el motor V8 permite a Node proporcionar un entorno de ejecución del lado del servidor que compila y ejecuta javascript a velocidades increíbles. El aumento de velocidad es importante debido a que V8 compila Javascript en

código de máquina nativo, en lugar de interpretarlo o ejecutarlo como bytecode. Node es de código abierto, y se ejecuta en Mac OS X, Windows y Linux.

Se utilizó el framework de ExpressJS, que es una infraestructura de aplicaciones web Node.js mínima y flexible que proporciona un conjunto sólido de características para las aplicaciones web y móviles. Se comunica con la API a través de Redis. También se trabaja con una base de datos en tiempo real llamada RethinkDB, que está diseñado para almacenar documentos JSON, y escalar a varias máquinas con muy poco esfuerzo. Cuenta con un lenguaje ameno y muy útil que permite la consulta entre uniones de tablas y grupos, fácil de configurar y aprender. Cuando un policía se conecta, se hace una petición a la api, y la api le manda la información a NodeJs, y éste lo agrega a la base de datos RethinkDB, que tiene funciones como por ejemplo encontrar al policía mas cercano cuando el usuario activa la alerta.

3.5 PANEL DE ADMINISTRACIÓN

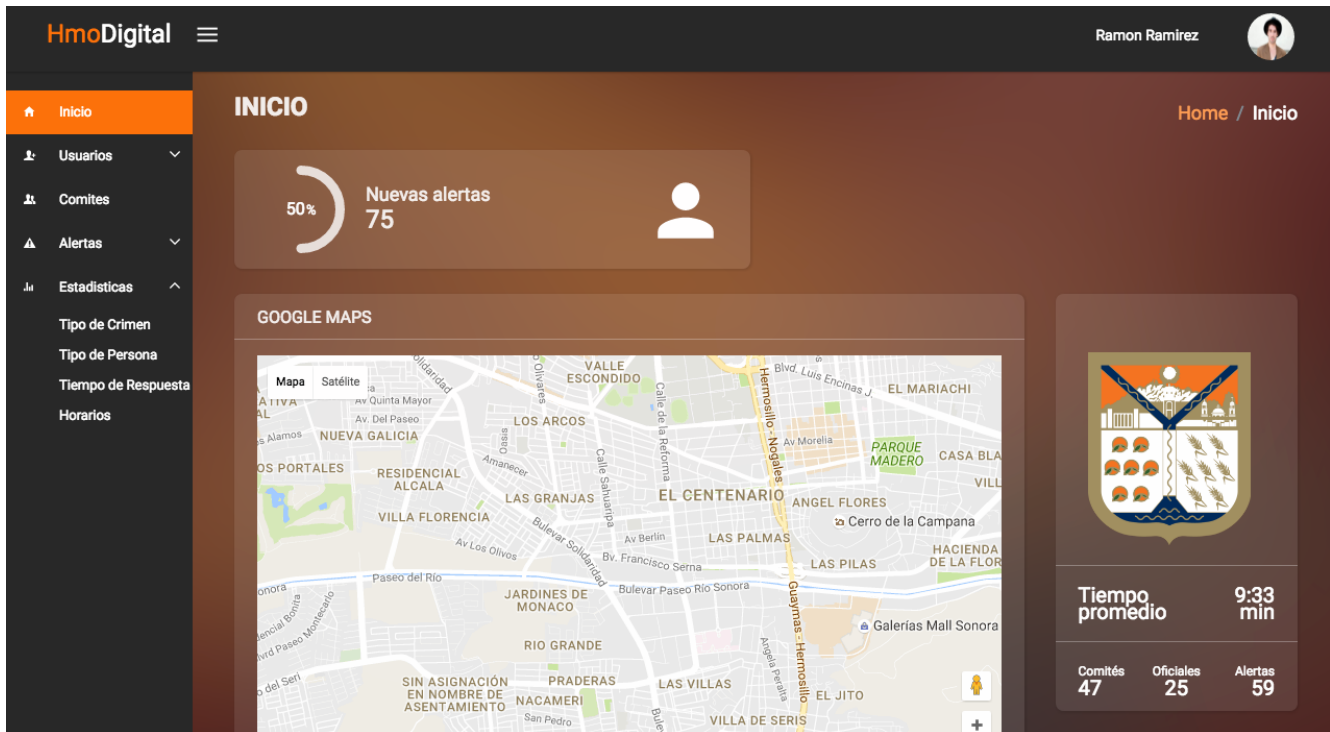


Figura 3.9. Panel de administración

El panel de administración se hizo en un framework de javascript llamado Angular 2, que es un framework para el Front End de la aplicación donde se trabaja con componentes. Ahorita éste framework está en Beta, pero es un proyecto muy estable, hay muchas personas manteniendo el código de Angular 2. Éste dashboard, no lo desarrollé no, lo desarrolló mi compañero de equipo, en la cual yo hice un plan de trabajo para el dashboard. Se implementó la metodología SCRUM para el desarrollo de éste dashboard. Se planeo para un mes y medio y efectivamente se cumplió con el tiempo.

4 DESCRIPCIÓN DETALLADA DE LAS ACTIVIDADES REALIZADAS

4.1 PRIMERA ETAPA

En la primera etapa, se planeó todo el desarrollo del sistema, se usaron diagramas de clases para las actividades de las 2 aplicaciones, posteriormente se hizo el diagrama de la base de datos, el cual fue cambiando mientras avanzaba el proyecto, ya que el cliente el cliente quería hacer cambios.

Después de haber terminado los diagramas y el esquema de la base de datos, se empezó a discutir que si que metodología íbamos a escoger, y la que ganó fue SCRUM, que es un proceso en el que se aplican de manera regular un conjunto de buenas prácticas para trabajar colaborativamente, en equipo, y obtener el mejor resultado posible de un proyecto. Estas prácticas se apoyan unas a otras y su selección tiene origen en un estudio de la manera de trabajar de equipos altamente productivos.

En Scrum se realizan entregas parciales y regulares del producto final, priorizadas por el beneficio que aportan al receptor del proyecto. Por ello, Scrum está especialmente indicado para proyectos en entornos complejos, donde se necesita obtener resultados pronto, donde los requisitos son cambiantes o poco definidos, donde la innovación, la competitividad, la flexibilidad y la productividad son fundamentales.

Scrum también se utiliza para resolver situaciones en que no se está entregando al cliente lo que necesita, cuando las entregas se alargan demasiado, los costes se disparan o la calidad no es aceptable, cuando se necesita capacidad de reacción ante la competencia, cuando la moral de los equipos es baja y la rotación alta, cuando es necesario identificar y solucionar ineficiencias sistemáticamente o cuando se quiere trabajar utilizando un proceso especializado en el desarrollo de producto.

El plan de trabajo lo desarrollé en una semana, se basó en todas las actividades desarrolladas en el diagrama de actividades y se planeó el desarrollo para 2 meses.

4.2 SEGUNDA ETAPA ETAPA

Ya que se implementó Scrum, se empezó a desarrollar las aplicaciones y los servicios web. Empecé yo solo el primer mes y medio, lo que hice fue empezar las 2 aplicaciones móviles en Ionic. También empecé el proyecto con Ruby on Rails, y el otro con Node Js.

Ya que tenía las vistas del login y crear usuario, empecé a desarrollar los modelos y controladores de Ruby On Rails. El modelo es el que tiene los atributos de una tabla para después hacer la migración y posteriormente el Active Record de Rails genera la consulta para crear dicha tabla en la base de datos. El controlador es el que modela la información para posteriormente enviarla a través de JSON cuando se hace una petición desde la aplicación. Para las rutas, se tienen que configurar en un archivo de rutas del framework.

Ya que se terminaron los servicios web, se empezó a consumir la api en la aplicación y se hizo testing; terminando se quitó dicha tarea en el tablero de Scrum. Aquí ya habían pasado 2 semanas.

Después se dedicó una semana a la aplicación de policía, lo que hice es el mapa de la aplicación (figura 3.1), instalando el plugin nativo de Google Maps. Cada vez que el policía inicia sesión manda una petición de "log in" a la Api en Ruby, y éste le manda los datos de policía a NodeJS y lo agrega a la base de datos en tiempo real. También hice que cada vez que la aplicación detecte un cambio de posición, mande una petición directamente a NodeJs para modificar su ubicación en la base de datos de tiempo real (RethinkDb).

A la cuarta semana se contrató a un integrante del equipo para desarrollar el panel de administración. Hice una planeación para el dashboard, el cual se trabajó desde la 4ta semana hasta la 8va semana.

En la cuarta, quinta y sexta semana se trabajó con la aplicación del comité para empezar hacer la lógica de la aplicación. Se empezó a hacer el botón de alerta, que lo que hace es mandar una petición georeferenciada a la Api, ésta hace un registro en la base de datos y posteriormente la manda a NodeJs para enviarla al policía más cercano utilizando los métodos de la base de datos en tiempo real. Se crea un cuarto de comunicación para que solo el policía y ese usuario estén conectados punto a punto, utilizando sockets, así el policía ya no puede recibir más alertas y sólo va a emitir eventos al usuario. Al llegar la notificación al policía, éste la acepta y manda una petición a la Api, modificando el estado de la alerta, que lo cambia a "En proceso", y por medio de sockets le manda un evento al usuario, y al usuario le aparece una pantalla donde le aparecen los datos del policía y se inicia un cronómetro que para hasta que el policía llegue a

atender. En éste proceso se pueden hacer llamadas y mandar mensajes, tanto del usuario al policía y viceversa.

Cuando el policía llega a la ubicación tiene que terminar la alerta, lo que hace es mandar una petición a la Api para actualizar el estado de la alerta a “atendida”, y a través de NodeJs, le manda un evento al usuario que ya se llegó a atender la alerta. Después de esto los 2 tienen que calificarse, si la alarma es falsa, el policía puede bloquear al usuario.

4.3 TERCERA ETAPA

En ésta etapa monté todo el servidor, rentamos uno a través de Digital Ocean, y cree un servidor con Ubuntu 14.04 y le instalé docker, git y MySQL. Cloné el proyecto de GitHub de la Api en Ruby on Rails para crear la imagen de Docker para su uso público.

Después instalé Redis y RethinkDb. En Rethink tuve que crear la base de datos y la tabla de policías y crear un índice para la geo localización para acceder a las funciones para encontrar al policía más cercano.

Después bajé el proyecto del tiempo real desde GitHub, y lo instalé como servicio para que corra en el servidor en dicho modo.

Ya que se terminó la aplicación en el tiempo establecido, se hizo una junta con el Presidente Municipal, el secretario del ayuntamiento y el equipo de ellos, para presentar la aplicación. Les gustó mucho, pero pidieron muchos módulos nuevos para 1 semana. Los módulos eran uno de chat, otro de las listas de vecinos y otro de calificaciones de policía a usuario y usuario a policía. Se hicieron efectivamente en una semana y se volvió a presentar el proyecto. Ya quedado conforme el cliente (presidente municipal), se empezó a capacitar a los policías. La capacitación la hice yo. Luego capacité a 5 capacitadores, los cuales capacitarán a los integrantes de los comités en 3 colonias en Hermosillo (Villa Bonita, La choya y Los Agaves). Ahorita se está dándole soporte al proyecto y haciendo más eficiente el código.

5 ANÁLISIS DE LA EXPERIENCIA ADQUIRIDA

En este apartado se presentan la valoración de los aprendizajes y lecciones aprendidas durante mi estancia profesional en la empresa QuantumBit, tanto en términos del proyecto en general, los objetivos que se plantearon para su realización, las actividades desarrolladas y la metodología que se utilizó.

5.1 ANÁLISIS GENERAL DEL PROYECTO

La empresa QuantumBit, es una empresa donde se relaciona directamente con el gobierno municipal. Se manejan proyectos muy grandes del nuevo gobierno. Aprendí mucho de desarrollo de sistemas completos, tanto del front end como el back end, también liderazgo de proyectos ya que yo analicé todo el sistema e hice el plan de desarrollo, y por último comunicación con el cliente, ya que en las juntas yo hablaba para presentar el proyecto y aprendí a ver como negocian entre empresas privadas y el gobierno.

5.2 ANÁLISIS DE LOS OBJETIVOS DE LAS PRÁCTICAS

El objetivo de mis prácticas profesionales, era de cumplir dicho requerimiento para el egreso, es diferente a mi objetivo profesional ya que ya tengo un año laborando en ésta empresa, y como objetivo tengo seguir aprendiendo de liderazgo de proyectos para luego poder administrar proyectos grandes, ya que es en lo que me estoy especializando. Mi primer objetivo antes de entrar a ésta empresa fue saber hacer sistemas completos, cumplí dicho objetivo, y posteriormente fije mi objetivo más alto.

5.3 ANÁLISIS DE LAS ACTIVIDADES REALIZADAS

Una de las ventajas de trabajar en ésta empresa es que siempre hay proyectos nuevos y diferentes, en el cual se requiere aprender cosas nuevas. Cuando entré solo sabía de desarrollo web, Front end. A los 6 meses me hice Full Stack Developer, el cual ya implica web services y vistas. Y mi objetivo es ser project manager.

Aprendí mucho de desarrollo de software, se como funcionan los servicios REST, aprendí a configurar servidores en modo de producción, aprendí a usar GitHub para el trabajo entre los desarrolladores del equipo, y por último aprendí a analizar y administrar proyectos.

6 CONCLUSIONES Y RECOMENDACIONES

Las prácticas profesionales se desarrollaron en una empresa relacionada con el gobierno, que se rige por normas y políticas establecidas por el dueño de la empresa y sus socios. Sin embargo, considero que se tienen algunas áreas de oportunidad que se puedan atender, entre ellas las siguientes:

- ❖ A la empresa le hace falta un administrador de proyectos de planta y un administrador de base de datos. El administrador de proyectos se necesita, por que son proyectos muy grandes y con fechas de entregas muy cortas, entonces éstos proyectos requieren de un buen análisis y administración para poderlo sacar a los tiempos establecidos y con los menos errores posibles.
- ❖ Se necesita un administrador de base de datos, por que así se le ahorra trabajo a los programadores en cada proyecto, entonces ya teniendo la estructura de la base de datos es más fácil poder hacer el desarrollo y la lógica del sistema.

En ésta empresa he aprendido muchas cosas, desde desarrollar una página web hasta desarrollar una aplicación móvil con todo y servicios web. También he aprendido lo que es analizar un proyecto, para después planear bien el desarrollo, también aprendí hacer diagramas de bases de datos, usar las relaciones entre tabas y sentencias SQL.

7 REFERENCIAS BIBLIOGRÁFICAS Y VIRTUALES

Sutherland, Jeff (2014), Scrum: The Art of Doing Twice the Work in Half the Time.

Blanchard, Benjamin (1995), Ingeniería en Sistemas.

StackOverFlow, www.stackoverflow.com, sitio web para preguntas y respuestas relacionadas a la programación. La usaba todos los días

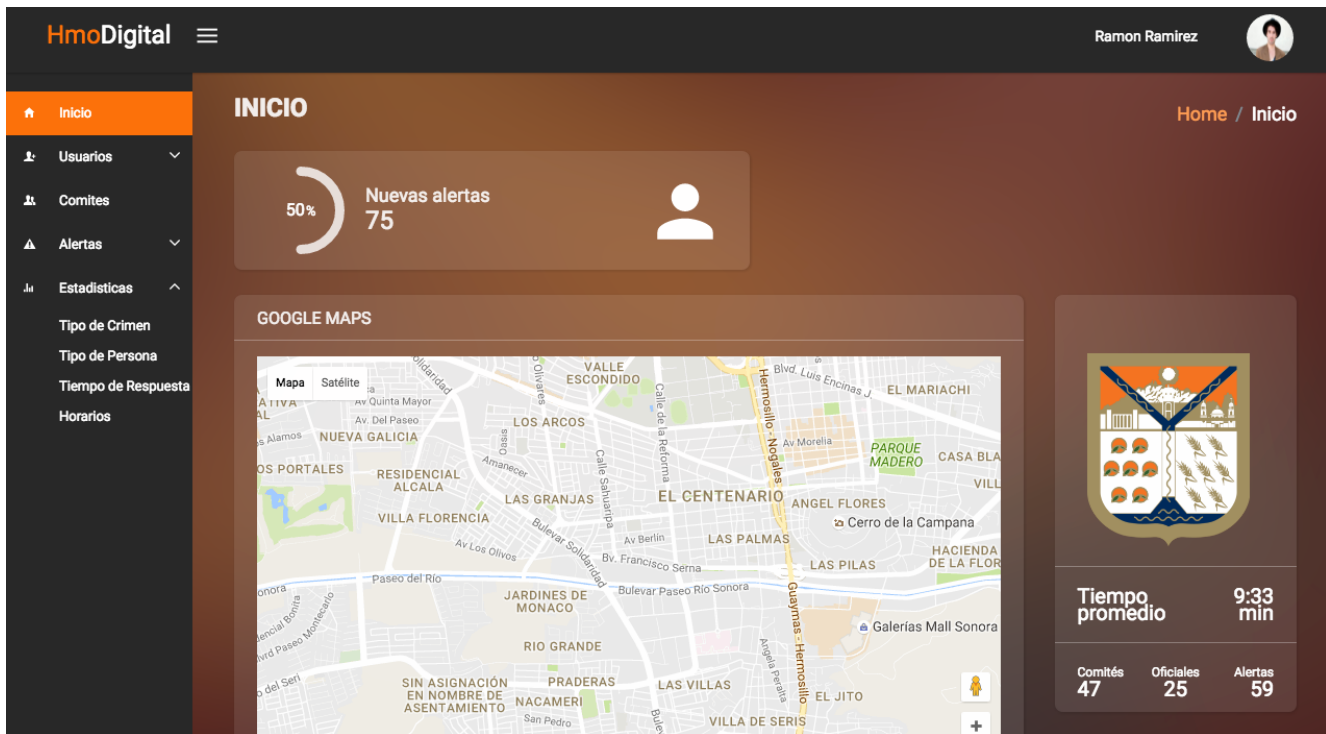
Ionic Docs, <http://ionicframework.com/docs/v2/>, Documentación oficial de Ionic. La usaba todos los días.

Angular2 Docs, www.angular.io, Documentación oficial de Angular 2. La usaba todos los días.

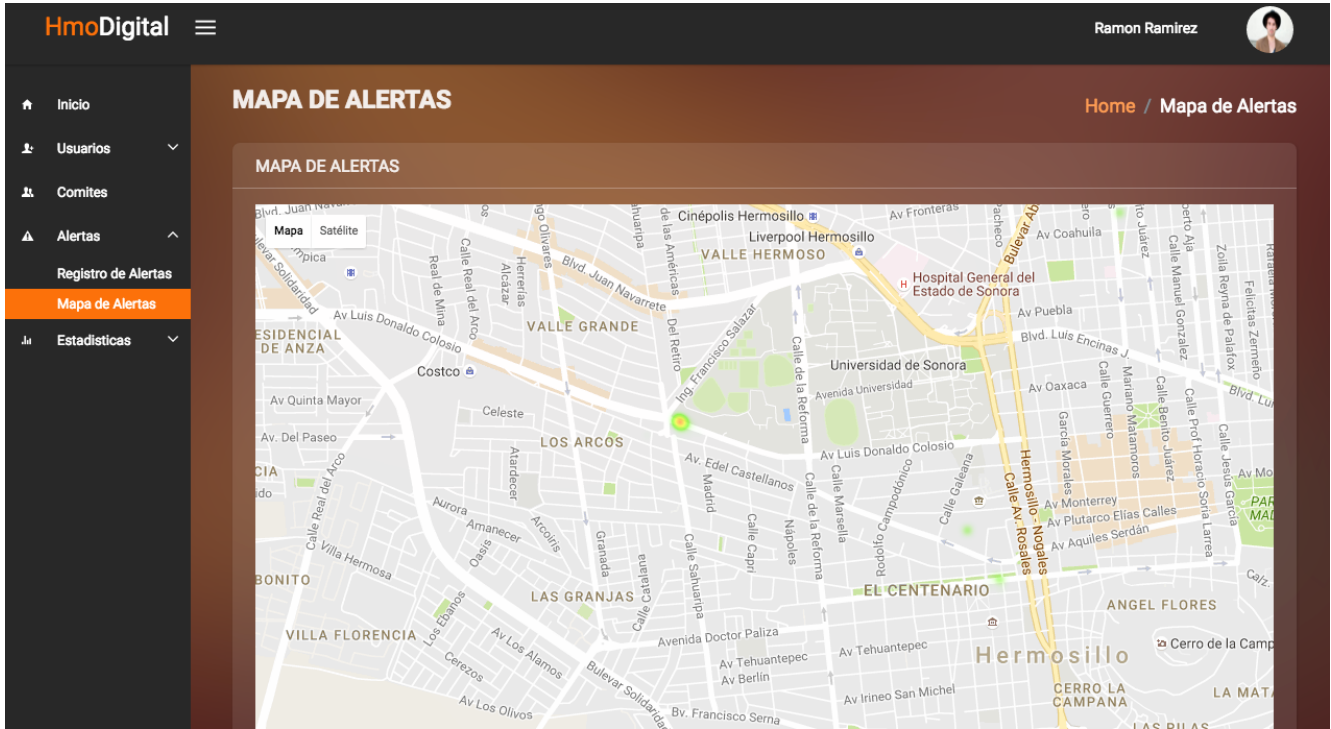
Crockford, Douglas (2008), JavaScript: The good parts.

Ruby on Rails Docs, <http://guides.rubyonrails.org/>, documentación oficial de Ruby on Rails.

ANEXOS



Anexo 1- Pantalla principal de panel de administración.



Anexo 2- Mapa de alertas

Anexo 3- Estadísticas de horarios de alertas.

