UNIVERSIDAD DE SONORA

DIVISIÓN DE INGENIERÍA
DEPARTAMENTO DE INGENIERÍA INDUSTRIAL
INGENIERÍA EN SISTEMAS DE INFORMACIÓN

REPORTE DE PRÁCTICAS PROFESIONALES:

NEARSOFT ACADEMY

PRESENTA:

Paulina Alejandra Pacheco Morales

Expediente: 209202584

ÍNDICE

1. INTRODUCCIÓN

Como uno de los requisitos para completar el proceso de titulación, la Universidad de Sonora solicita a los estudiantes realizar prácticas profesionales dentro de una empresa o institución donde puedan desarrollar y aplicar los conocimientos adquiridos a través de los cursos que se le fueron impartidos a lo largo de su carrera. El estudiante debe laborar dentro del organismo de su elección (y que sea aprobado por la Universidad de Sonora) por un tiempo total mínimo de 340 horas, con duración mínima de 8 semanas.

Como apoyo para los alumnos, la Universidad cuenta con medios para registrar solicitudes en empresas de distintas áreas laborales que requieran la ayuda de uno o varios practicantes o que deseen brindarle a un estudiante interesado la experiencia de crecimiento laboral durante su estancia en la empresa. De igual manera el alumno puede realizar su propia propuesta y presentar una solicitud para realizar las prácticas profesionales en una empresa de su decisión, la cual tiene que ser aprobada por el encargado de prácticas profesionales de su departamento.

1.1 Objetivo del proyecto de prácticas profesionales

Con la realización de las prácticas profesionales, la Universidad de Sonora pretende que el alumno adquiera en cierta medida experiencia en el campo laboral y desarrollar habilidades y aptitudes que le permitan al alumno complementar los conocimientos adquiridos en las clases. De igual manera, la Universidad adquiere el reconocimiento de las empresas locales y se asegura de que estas consideren a los alumnos de la Universidad de Sonora como fuente de candidatos competentes que puedan satisfacer las necesidades de la compañía.

1.2 Proceso de admisión de Nearsoft Academy

En esta ocasión se contactó a la empresa Nearsoft para solicitar la estadía como practicante en la empresa. Ellos cuentan con un programa llamado Nearsoft Academy el cual reúne las características necesarias para ser considerado para completar las prácticas profesionales.

Se comenzó el proceso de reclutamiento, el que consiste en tres partes: logic test, entrevista en inglés y screening; cada uno tiene que ser aprobado antes de pasar al siguiente, de otra manera el aspirante no es aceptado dentro del programa. El logic test consiste en un examen online que presenta una serie de reglas con las cuales tienes que ir respondiendo las preguntas de lógica. A continuación, se realiza una entrevista en inglés: una llamada de voz con uno de los integrantes de Nearsoft para valorar si el aspirante tiene un nivel de inglés conversacional. Por último, se realiza una entrevista en persona con Isaac López, encargado del programa de Nearsoft Academy, quien valora las aptitudes del aspirante mediante preguntas técnicas y retóricas, aparte de un pequeño ejercicio para saber la capacidad de resolver problemas.

1.3 Primeros días en la empresa

Después de haber pasado por este proceso de aceptación, se le solicitó a la practicante empezar sus actividades en la oficina a partir de enero 15 de 2016. En este período se realizó una introducción a un sistema interno de la empresa que sirve para los reclutadores en su proceso de entrevistas de tal manera que el practicante pueda familiarizarse con él y trabajar en mejoras en el código, así como la resolución de problemas encontrados en el mismo. Además de esto, el practicante colaboró en otro proyecto que consistió en adaptar unas plantillas o diseños elaborados en Photoshop y convertirlas a HTML y CSS con la ayuda de Yeoman, una herramienta utilizada para construir aplicaciones web, la cual cuenta con la opción de integrar Sass (framework de CSS) al proyecto creado.

El programa de Nearsoft Academy, generación 2016-1, empezó formalmente el día 8 de febrero de 2016 y terminó en junio del mismo año.

1.4 Acerca del reporte de prácticas profesionales

El presente trabajo es una memoria de las actividades que se realizaron durante este tiempo en la empresa. En primer lugar, se hará una descripción del área del trabajo. En el segundo capítulo se definirá el proyecto del cual fue parte el practicante. Seguidamente se hará una revisión de conceptos necesarios para comprender los temas tratados en los siguientes capítulos. En el capítulo cinco se habla de cómo se desarrolló el proyecto y las actividades realizadas por el practicante. Para finalizar, en el capítulo seis se hace un análisis de cada fase del programa y los aprendizajes obtenidos y en el capítulo final se incluyen las conclusiones y recomendaciones por parte del practicante.

2. Descripción de la empresa

Nearsoft Inc. ofrece a sus clientes alternativas y soluciones en tecnología basadas en el desarrollo de sistemas web y otros servicios. La empresa ayuda a compañías que residen en Estados Unidos a extender su equipo de trabajo con ingenieros latinoamericanos, principalmente de México. Cuenta actualmente con tres oficinas: en Ciudad de México, Chihuahua y la principal en Hermosillo, Sonora.

Figura 2-1 Logo de Nearsoft

Su filosofía se basa principalmente en ser una empresa que promueva la innovación, y esto se refleja en su jerarquía horizontal: una estructura horizontal es una sin mandos intermedios donde se les permite a los empleados tomar sus propias decisiones operativas día a día. También se conoce como una organización "plana" [1]. Gracias a esto se destaca entre el modelo tradicional de jerarquía vertical, lo cual les da a los empleados la libertad y confianza de ofrecer propuestas para la mejora de las actividades dentro de la empresa.

2.1 Modo de trabajo dentro de Nearsoft

En Nearsoft no hay horarios de trabajo ni código de vestimenta, tampoco es necesario estar presente en la oficina para trabajar. El empleado tiene que asegurarse de cumplir el trabajo equivalente a 40 horas a la semana, sin embargo, no está bajo supervisión directa de nadie. Normalmente los empleados son designados a un cliente en particular, según la posición que tomen dentro de la empresa. Entre estos equipos se pueden mencionar a Switchfly, SkyTouch, Points y MDLive.

La mayoría de estos equipos tienen un estilo de trabajo en el que se utilizan metodologías ágiles como SCRUM. Las metodologías ágiles son aquellas que permiten adaptar la forma de trabajo a las condiciones del proyecto, consiguiendo flexibilidad e inmediatez en la respuesta para amoldar el proyecto y su desarrollo a las circunstancias específicas del entorno. En esencia, las empresas que apuestan por esta metodología consiguen gestionar sus proyectos de forma eficaz reduciendo los costes e incrementando su productividad [2]. Los equipos de trabajo utilizan artefactos como los daily meetings, donde se informa el avance que se tuvo el día anterior y los bloqueos que pueda tener en ese momento.

2.2 Iniciativas de la empresa

Además del trabajo que ofrece a sus clientes, Nearsoft también se encarga de ofrecer eventos que ayuden a la comunidad a involucrarse en temas de tecnología e innovación. Nearsoft forma parte de la mayoría de las comunidades de software en México, creando nuevos eventos y apoyando a algunos que ya existen. Entre ellos se pueden mencionar Node School, Android school, Django Girls, Dev Circles, JVMX entre otros.

Figura 2.2-1 Nearsoft es anfitrión de muchas comunidades de tecnología en México

Además de eventos externos, Nearsoft invita a cualquiera que desee a acudir a sesiones de aprendizaje que se imparten dentro de las oficinas de Hermosillo. Nearsoft Apprentice es un programa dirigido a cualquier persona que tenga deseos de aprender, desde estudiantes hasta profesionistas de otras áreas que deseen adentrarse en el mundo del desarrollo de software. Se discuten temas como base de datos, ciencia de la computación, métodos ágiles y diseño de software. En las sesiones teóricas se definen conceptos usados en las ciencias computaciones y en las sesiones prácticas se aplican estos conceptos en código y ejemplos reales.

Por último, se encuentra otra iniciativa: el programa de Nearsoft Academy está dirigido a estudiantes recién graduados o que se encuentren cursando los últimos semestres en carreras relacionadas a las tecnologías computacionales. Es una oportunidad para que los participantes aprendan las buenas prácticas junto con la ayuda de mentores experimentados.

3. Descripción del proyecto

Este programa (Nearsoft Academy) surgió de la idea de apoyar al talento joven a adquirir buenas prácticas y experiencia para mejorar su currículum, y a la vez preparase para entrevistas con los clientes de Nearsoft con necesidades de alta calidad en software. Esta iniciativa tiene planeada una duración de seis meses, durante los cuales los participantes estarán trabajando bajo la supervisión de un mentor en el objetivo personal definido por ellos: ya sea especializarse en alguna tecnología en específico o aplicar a una de las vacantes disponibles.

3.1 Objetivos generales

- Ofrecer algo a cambio a la comunidad de código abierto.
- Entrenar a futuros emprendedores de la tecnología.
- Proveer un ambiente de práctica y que se comparta el conocimiento.

3.2 Misión

- Entrenar a través de la práctica.
- Desarrollar el talento del practicante.
- Ayudar a los candidatos a construir un portafolio de logros tangibles.
- Ayudar a las universidades a que sus estudiantes emprendan en el mundo laboral.

De esta manera, la empresa obtendrá

- Nuevos posibles empleados competentes.
- Convertirse en una fuente de conocimiento para la comunidad.
- Ayudar a expandir la comunidad emprendedora de tecnología en México y Latinoamérica.

3.3 Objetivos y regulaciones

- Nearsoft no es una institución educacional y no es su propósito convertirse en una [3].

- El propósito es que los integrantes aprendan a través de la práctica.
- Para hacerlo escalable, se está experimentando con el uso de proyectos de código abierto (Open Source) como una tarea común durante la duración del internship. Los participantes pueden unirse a uno de estos proyectos (que sean de su elección, siempre y cuando cumpla con los requisitos) y finalmente convertirse en un contribuidor aceptado dentro de la comunidad del proyecto.
- Además de su participación en proyectos de open source, los candidatos deben realizar presentaciones, escribir posts, analizar y explicar código, entre otras actividades.
- Como una manera de analizar su aprendizaje del idioma inglés, se invita a los candidatos a participar en juntas que sean completamente en inglés.

3.4 Alcances y limitaciones

El alcance principal en la participación de este proyecto es culminar con el contrato en la empresa para que el participante se convierta en empleado de esta. Sin embargo, existen otros alcances de menor magnitud como:

- Reparar errores y bugs de proyectos de open source,
- Desarrollar nuevas features en los sistemas de Nearsoft,
- Crear publicaciones (posts) que hablen de algún tema relacionado a tecnología,
- Dar pláticas sobre temas concernientes al desarrollo de software.

Entre las limitaciones del proyecto se pueden encontrar que:

- Solo se aceptan proyectos de open source activos y actualizados.
- Para entrar al programa, los candidatos deben de haber estado graduados recientemente o estar cursando el último semestre, además de aprobar el logic test y la entrevista en inglés.
- Una vez aceptados, se hace una evaluación para determinar las fortalezas y debilidades para guiar su entrenamiento.
- Se realizan cheques regulares y formales guiados por la evaluación inicial.

3.5 Estructura del proyecto

El programa de Nearsoft Academy se centra principalmente en hacer que el candidato aprenda las habilidades necesarias para convertirse en un desarrollador a través de la práctica, además de mejorar su desempeño general basado en el fortalecimiento de sus cualidades y ayudar al candidato a construir una reputación a través de los aportes generados dentro de la empresa y en las comunidades de código abierto para abrir las oportunidades de empleo en Nearsoft u otras organizaciones que ofrezcan un puesto como desarrollador y similares.

Para lograr con sus objetivos y metas, los encargados del programa han establecido que éste se irá desenvolviendo en cinco fases, cada una enfocada en diferentes actividades que ayudan al candidato a integrarse de manera efectiva a las tareas de la empresa y sus labores.

Las fases en las que se divide el programa de Nearsoft Academy son la fase de *Reset*, fase *Team*, fase *Open Source*, fase *Real client/users* y por último la fase *Get a job.*

En figura 3.5-1 se explica brevemente en qué consiste cada una de las fases que conforman al Nearsoft Academy. En el capítulo cinco se describirán de manera más detallada las actividades que se realizaron en cada una de ellas.

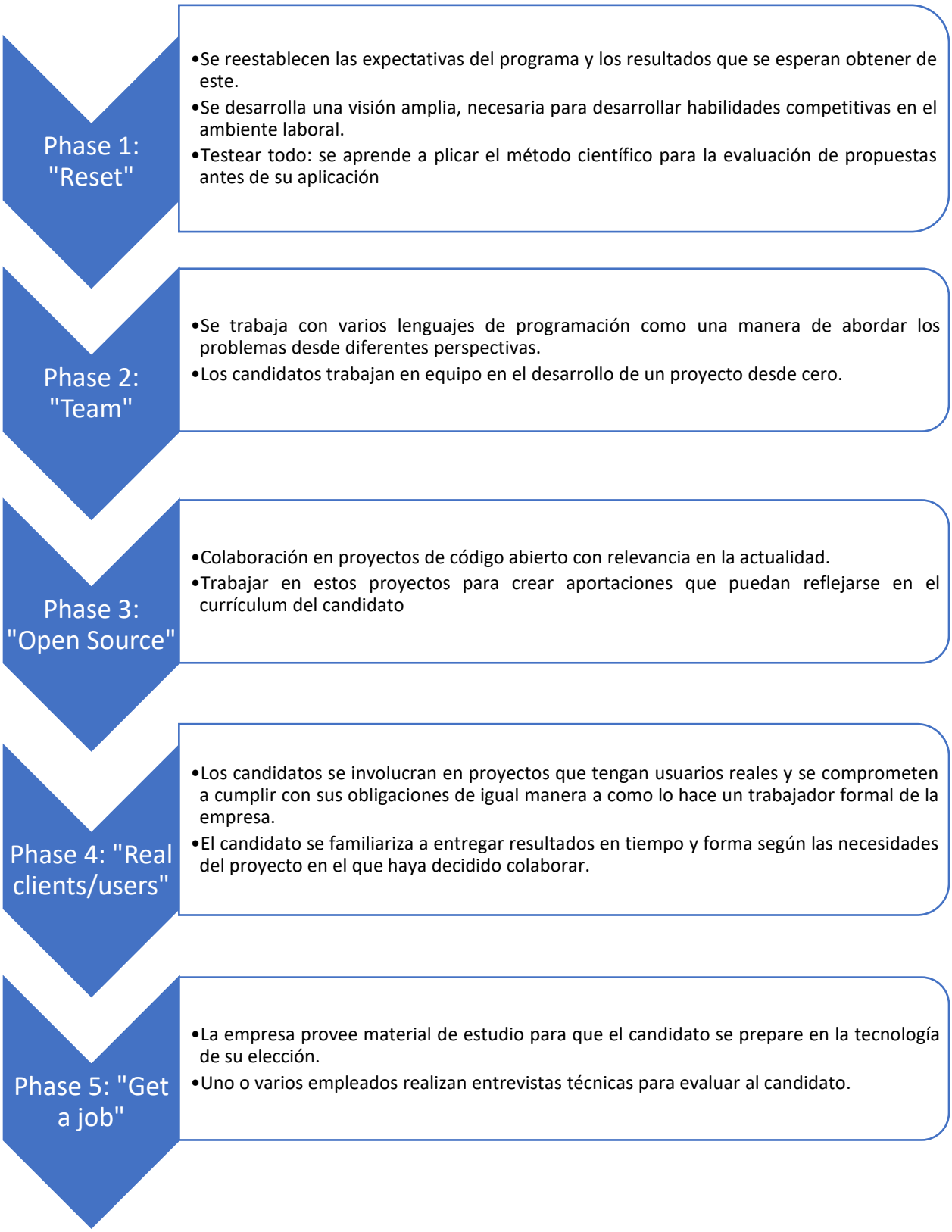

Figura 3.5-1 Breve descripción de las fases que conforman a Nearsoft Academy

4. Fundamento teórico de las herramientas y conocimientos aplicados

Para el mejor entendimiento de los próximos capítulos, se definirán algunos conceptos útiles para la comprensión de las actividades realizadas durante este proyecto y se hará una pequeña introducción de las herramientas que se utilizaron durante este período.

En primer lugar, se definirán algunos conceptos técnicos utilizados en el desarrollo de software, después los lenguajes de programación usados y por último frameworks y herramientas.

4.1 Conceptos y técnicas

4.1.1. Sistemas Web

El objetivo principal de una aplicación o sistema web es permitir al usuario realizar una tarea dentro de este sistema, normalmente haciendo el uso de un navegador web. Para desarrollarlos, se utilizan lenguajes web como Javascript y otras herramientas como CSS y HTML. Hoy en día es muy común que los desarrolladores utilicen frameworks basados en diferentes lenguajes de programación, los cuales ayudan a que la creación de estos sistemas se dé de manera más rápida y ágil.

El sistema web se diseñó para ser de modo cliente/servidor. Eso quiere decir que hay un elemento activo, en este caso el navegador web, que haría las veces de cliente, y un elemento pasivo, en este caso el servidor web, que haría las veces de servidor [4].

4.1.2 Servidor Web

Es el programa que se mantiene a la escucha para atender las peticiones de los navegadores, a modo de servir las páginas web. También puede ejecutar códigos para generar páginas, imágenes y otros contenidos para cada petición. Hay servidores de pago y gratuitos, y libres, algunos ejemplos de los más usados: Apache, Cherokee y Microsoft IIS.

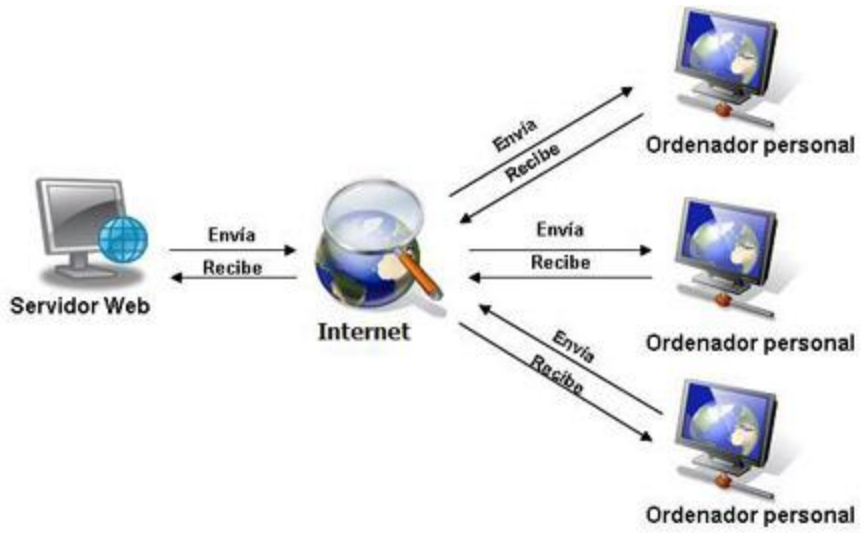

Figura 4.1.2-1 Diagrama que demuestra cómo funciona un servidor web

4.1.3 SCRUM

El Scrum es un proceso de la Metodología Ágil que se usa para minimizar los riesgos durante la realización de un proyecto, pero de manera colaborativa.

Figura 4.1.3-1 Proceso de la metodología ágil SCRUM

Entre las ventajas se encuentran la productividad, calidad y que se realiza un seguimiento diario de los avances del proyecto, logrando que los integrantes estén unidos, comunicados y que el cliente vaya

viendo los avances. La profundidad de las tareas que se asignan en SCRUM tiende a ser incremental, caso que coincide exactamente con el devenir normal de un desarrollo [5].

Es perfecto para empresas de desarrollo de software orientadas a varios clientes.

4.1.4 Programación Orientada a Objetos

Un lenguaje orientado a objetos es un lenguaje de programación que se centra en el diseño de aplicaciones orientadas a elementos llamados objetos.
Podemos definir un objeto como un conjunto complejo de datos y programas que poseen estructura y forman parte de una organización.

Esta definición especifica varias propiedades importantes de los objetos. En primer lugar, un objeto no es un dato simple, sino que contiene en su interior cierto número de componentes bien estructurados. En segundo lugar, cada objeto no es un ente aislado, sino que forma parte de una organización jerárquica o de otro tipo. Todos los objetos poseen tres partes importantes: relaciones, propiedades y métodos.

Las relaciones permiten que el objeto se inserte en la organización y están formadas esencialmente por punteros a otros objetos. Las propiedades distinguen un objeto determinado de los restantes que forman parte de la misma organización y tiene valores que dependen de la propiedad de que se trate. Las propiedades de un objeto pueden ser heredadas a sus descendientes en la organización. Los métodos son las operaciones que pueden realizarse sobre el objeto, que normalmente estarán incorporados en forma de programas (código) que el objeto es capaz de ejecutar y que también pone a disposición de sus descendientes a través de la herencia [6].

4.1.5 Programación funcional

Los lenguajes de programación funcionales son aquellos lenguajes donde las variables no tienen estado — no hay cambios en éstas a lo largo del tiempo — y son inmutables — no pueden cambiarse los valores a lo largo de la ejecución. Además, los programas se estructuran componiendo expresiones que se evalúan como funciones [7].

En los lenguajes funcionales las instrucciones cíclicas como for, while y do-while no existen. Todo se procesa usando recursividad y funciones de alto orden.

4.2 Lenguajes de programación y protocolos web

A lo largo de todo este proyecto, se utilizaron distintos lenguajes de programación, cada uno con diferentes fines.

4.2.1 Java

Java es un lenguaje de programación y una plataforma informática utilizada en la creación de software. Este lenguaje está basado en el paradigma de programación orientada a objetos.

Para poder ejecutar código escrito en Java, se necesita instalar Java Runtime Enviroment (JRE) en el ambiente. JRE está formado por Java Virtual Machine (JVM), clases del núcleo de la plataforma Java y bibliotecas de la plataforma Java de soporte. JRE es la parte de tiempo de ejecución del software de Java, que es todo lo que necesita para ejecutarlo en el explorador web [4]. Fue diseñado específicamente para tener tan pocas dependencias de implementación como fuera posible. Su intención es permitir que los desarrolladores de aplicaciones escriban el programa una vez y lo ejecuten en cualquier dispositivo (conocido en inglés como WORA, o "write once, run anywhere"), lo que quiere decir que el código que es ejecutado en una plataforma no tiene que ser recompilado para correr en otra [5].

4.2.2 Javascript

Javascript es un lenguaje de programación ampliamente utilizado en el desarrollo web que surgió por la necesidad de ampliar las posibilidades del HTML. Este lenguaje permite a los desarrolladores crear acciones en sus páginas web. JavaScript es un lenguaje interpretado línea a línea por el navegador, mientras se carga la página, que solamente es capaz de realizar las acciones programadas en el entorno de esa página HTML donde reside. Sólo es posible utilizarlo con otro programa que sea capaz de interpretarlo, como los navegadores web.

Este es un lenguaje orientado a objetos, es decir que la mayoría de las instrucciones que se emplean en los programas, en realidad son llamadas a propiedades y métodos de objetos del navegador, y en algunos casos del propio lenguaje. En Java, en cambio, no hay nada que no esté en un objeto [6].

4.2.3 Elixir

Elixir es lenguaje de programación de propósito general, concurrente; este lenguaje es funcional. Además está construido sobre la máquina virtual de Erlang. Se trata de un lenguaje dinámico con una sintaxis flexible y apoyado en macros; que aprovecha las capacidades de Erlang para construir aplicaciones concurrentes y distribuidas, tolerables a fallos, con actualizaciones [11].

Elixir también soporta pattern matching, el polimorfismo a través de protocolos (similar a Clojure), alias y estructuras de datos asociativos (generalmente conocido como hashes en otros lenguajes de programación).

Por último, Elixir y Erlang comparten el mismo byte code. Esto significa que puede invocar código Erlang de Elixir (y viceversa) sin ningún tipo de transformación o impacto en el rendimiento. Esto permite a los desarrolladores mezclar la expresividad de Elixir con la robustez y el rendimiento de Erlang.

4.2.4 HTML y CSS

HTML es el lenguaje con el que se define el contenido de las páginas web. Básicamente se trata de un conjunto de etiquetas que sirven para definir el texto y otros elementos que compondrán una página web, como imágenes, listas, vídeos, etc. Se podría decir que HTML es el esqueleto que define la estructura de una página web.

CSS o Cascading Style Sheets, traducido literalmente al español, como *Hojas de estilo en cascada*, es un lenguaje para la especificar cómo los documentos se presentan a los usuarios. El CSS se utiliza para aplicar estilos y diseños a los elementos HTML de una página web.

4.3 Frameworks y herramientas

Un framework es una estructura conceptual y tecnológica de soporte definido, normalmente con artefactos o módulos de software concretos, que puede servir de base para la organización y desarrollo de software. Vamos, una manera de hacernos más fácil la programación.

Las razones para utilizar un framework son evitar escribir código repetitivo, utilizar buenas prácticas de programación y desarrollar más rápido. A continuación, se describen algunos de los frameworks utilizados en este período.

4.3.1 NodeJS

Este framework basado en Javascript se utiliza en la creación y desarrollo de servidores para aplicaciones web. La meta número uno declarada de Node es "proporcionar una manera fácil para construir programas de red escalables".

Node.js es una librería y entorno de ejecución de E/S dirigida por eventos y por lo tanto asíncrona que se ejecuta sobre el intérprete de JavaScript creado por Google V8. provechando el motor V8 permite a Node proporciona 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 [12].

4.3.2 Gulp

Gulp es una herramienta, en forma de script en NodeJS, que te ayuda a automatizar tareas comunes en el desarrollo de una aplicación, como pueden ser: mover archivos de una carpeta a otra, eliminarlos, minificar código, sincronizar el navegador cuando modificas tu código, validar sintáxis, entre otras tareas.

4.3.3 SASS

Sass es un framework de CSS o un lenguaje de scripting (SassScript escrito en Ruby) que se compila a CSS. SASS necesita un programa compilador como Compass, Bourbon, o Susy para funcionar. Sass permite utilizar funciones que no existen en la CSS como variables, anidación, métodos, herencia y otras cosas.

4.3.4 Git

Git es una herramienta para la administración de versiones de software. La principal diferencia entre Git y cualquier otro VCS (Subversion y compañía incluidos) es cómo Git modela sus datos. Conceptualmente, la mayoría de los demás sistemas almacenan la información como una lista de cambios en los archivos. Estos sistemas (CVS, Subversion, Perforce, Bazaar, etc.) modelan la información que almacenan como un conjunto de archivos y las modificaciones hechas sobre cada uno de ellos a lo largo del tiempo [13].

5. Procedimientos empleados y actividades desarrolladas

El programa de Nearsoft Academy se conforma de cinco fases. Estas fases han ido cambiando con cada generación de internos y va sufriendo de modificaciones según los resultados obtenidos de la generación anterior o por otras circunstancias que estén sucediendo dentro de la empresa. Cabe mencionar que no todos los candidatos tienen que cursar todas las etapas si han sido contratados prematuramente como empleados del lugar.

En este capítulo se describirá la experiencia personal del participante, basado en la estructura del programa que se tenía durante ese momento. Cabe mencionar que, por motivos de la administración de la empresa y otros factores, durante esta generación no se trabajó en la fase 4 (real users/clients) por lo que no se reportarán procedimientos ni resultados del mismo.

5.1 Reset Phase

“Reset” es el nombre de la primera etapa del programa y tiene una duración de un mes. El objetivo de esta etapa es hacer ver a los integrantes “internos” el verdadero conocimiento que ellos tienen en comparación con profesionales que ya laboran en la empresa y que se den cuenta de lo mucho que les falta por aprender. Entre las actividades de esta etapa se encuentran:

- Videos con temáticas de interés en tecnología,
- Lecturas de libros y artículos (Pragmatic programmer, Resonate),
- Cursos (Big-O notation, algoritmos, técnicas de aprendizaje),
- Ejercicios de programación

Estas actividades son asignadas semanalmente con un tema distinto y al terminar la semana los internos deben elaborar un ensayo en el que se refleje lo que prendieron, totalmente escrito en inglés. Estos reportes son enviados por correo a los mentores del practicante y a los encargados del programa. El practicante recibió retroalimentación para cada uno de sus ensayos por parte de los recipientes.

Videos

Estos videos cubren una amplia variedad de temas entre ellos algunos hablan sobre como presentar ideas a un público de manera efectiva, algoritmos de comprensión utilizados en la vida real, estudios realizados a través de la historia, etc.

Tienen como finalidad proveer conocimientos sobre distintos temas y mostrar las dificultades en las que se han enfrentado otros en el pasado y como fueron superadas.

Lecturas

Durante esta etapa se leyeron distintos artículos y libros entre los cuales se encuentra “Pragmatic programmer” de Andy Hunt y Dave Thomas. Este libro habla sobre como un programador debería de pensar y actuar para poder desarrollar software que sea escalable, mantenible y construido bajo las buenas prácticas de la programación.

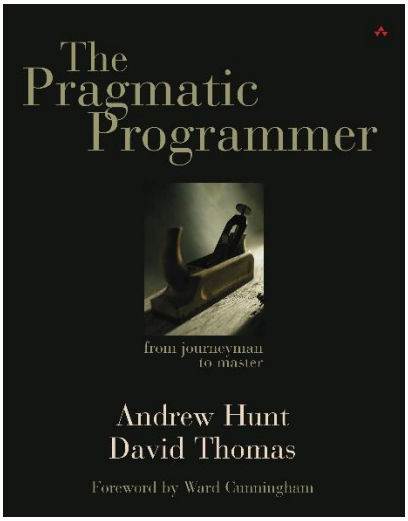

Figura 5.1-1 Portada de libro Pragmatic Programmer de Andrew Hunt

“Resonate” de Nancy Duarte fue otra de las principales lecturas en esta etapa en donde se explica cómo presentar una idea a un público de manera adecuada para captar su atención y generar el efecto deseado.

Cursos

Durante esta etapa se recibió un curso que constaba de sesiones diarias donde se habló sobre conceptos básicos de programación como SOLID y de orientado a objetos tales como herencia, polimorfismo, encapsulamiento y abstracción. Otros temas de interés fueron complejidad (big-O notation) y algoritmos de programación. Estas clases complementaban los temas vistos por el practicante dentro de sus lecturas y videos temáticos.

5.2 “Team” Phase

La tendencia natural de muchos de los participantes de Nearsoft Academy es comenzar a trabajar de manera solitaria: prefieren investigar por sí mismos y crear por ellos mismos, sin ayuda de nadie. Incluso durante la primera fase, el programa pretende que los mismos participantes descubran que unos de los objetivos principales es que se trabaje en equipo durante toda su estadía.

Afortunadamente, en esta generación los participantes tuvieron la iniciativa de trabajo en equipo desde el principio. En esta etapa se intenta promover aún más esta práctica para que forme parte de su vida laboral. En la primera parte de esta fase, se le asignó al practicante algunos ejercicios de programación. La segunda parte consistió en la elaboración de un proyecto totalmente nuevo.

5.2.1 Ejercicios de programación

Una de las tareas de esta fase fue la resolución de varios ejercicios de programación. Tres de ellos fueron parte de un evento pasado realizado por Google, llamado Google Code Jam. El practicante tuvo que investigar cómo resolver los problemas en tres diferentes lenguajes: Elixir, F# y Javascript.

El practicante, junto con sus compañeros del Academy, realizaron juntas en las cuales discutieron la resolución de los problemas y escribieron algoritmos en pseudo lenguaje. Posteriormente se crearon tres equipos diferentes: cada uno se enfocaría en resolver los tres problemas en un lenguaje en específico.

Después de tener lista la resolución de los ejercicios, se realizó una última junta para la revisión de los ejercicios y posibles mejoras al código realizado por los compañeros. Los tres ejercicios fueron resueltos

con éxito y fueron subidos a un repositorio de github para la aprobación por parte de los encargados del programa.

ReverseWords.fs Raw

```
open System
open System.IO

let read = File.ReadAllText("B-small-practice.in")
let lines = read.Split [|'\n'|]

for i in lines do
  let rev = Array.rev (i.Split [|' '|])
  let words = String.concat " " rev
  printf "%s" words
  printf "%c" '\n'
```

Figura 5.2.1-1 Solución al problema de Reverse words en F#

ReverseWords.exs Raw

```
raw = File.read! "B-small-practice.in"

list = String.split(raw, "\n")
list_wthoNo = List.delete_at(list, 0)

list_wthoNo |> Enum.with_index |> Enum.each( fn {n, i} ->
  words = Enum.reverse(String.split(n, " "))
  IO.puts "Case #"<>Integer.to_string(i+1)<>": "<>Enum.join(words, " ")
end)
```

Figura 5.2.1-2 Solución al problema de Reverse Words en Elixir

Se creó también un recomendador de películas (fig 5.2.1-3) basado en un archivo de 8 millones de reviews, el propósito fue generar un listado de películas para tres usuarios dados, basándose en las películas previamente vistas, para esto se utilizó big data, data mining, Mahout, Maven y unit testing.

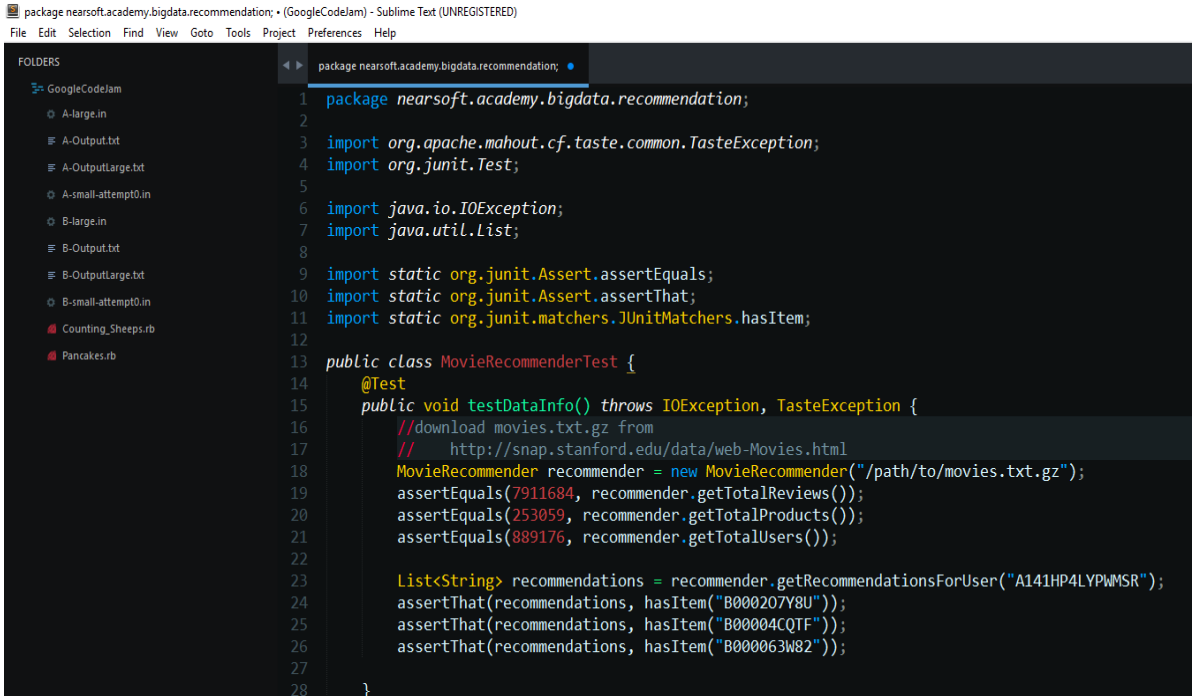

Figura 5.2.1.-3 Movie Recommender utilizando Mahout, Maven y Unit Testing

5.2.2 Construcción de un proyecto desde cero

En Nearsoft la cultura es un elemento muy importante, es considerada uno de los principales pilares de la empresa. Sin embargo, las oficinas de la empresa se encuentran localizadas en diferentes ciudades, por lo que mantener una cultura homogénea entre los trabajadores es una tarea difícil.

Para solucionar este problema, se les pidió a los integrantes del Academy construir un servicio que consistiera en mantener una video llamada permanente entre las tres oficinas de Nearsoft, pero con las medidas necesarias para que los usuarios sintieran que están viendo a la gente del otro lado a través de una ventana. En otras palabras, se precisaba que la comunicación se diera de manera natural, sin la sensación de estar utilizando ningún tipo de tecnología.

El proyecto que el practicante desarrolló está basado en el concepto de telepresencia. En un estudio de telepresencia el objetivo es despertar entre los usuarios la ilusión de que se encuentran conferenciando con sus interlocutores distantes sentados ante una mesa real [14] .

En un principio se idealizó el diseño de este proyecto como una ventana por la cual los interlocutores se verían entre ellos, por lo que llamamos se le llamó proyecto Kitchen Window, este diseño constó de un marco de ladrillos, un marco más pequeño que cubriría una pantalla, y tendría una repisa para colocar plantas o cualquier otro objeto decorativo para la simulación de la ventana. El streaming sería reproducido por una Raspberry Pi y una cámara digital conectada a dicho dispositivo, teniendo corriendo el streaming en una red privada con un ancho de banda estático y dedicado entregado por los servidores de Nearsoft, en la siguiente figura podemos observar el prototipo del dispositivo.

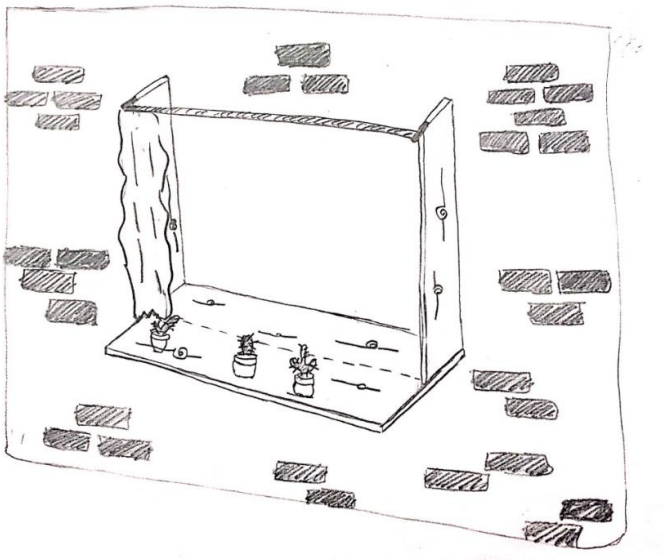

Figura 5.2.2-1 Bosquejo principal de Kitchen Window

Los integrantes del Academy decidieron organizarse en equipos más pequeños: un equipo de desarrollo de software, otro de hardware, un equipo de diseño y por último uno de documentación. El practicante en cuestión colaboró principalmente en el desarrollo del software, aunque también participó en parte del diseño del proyecto.

Se decidió trabajar bajo la metodología ágil SCRUM. Se sostuvieron juntas diarias en las que se reportaba el avance del día anterior de cada uno de los integrantes de los equipos. También se establecieron las historias de los sprint y se agendaron entregas semanales con el cliente (que en esta ocasión fueron los encargados de Nearsoft Academy quienes evaluaban estas entregas).

Posteriormente, el practicante hizo la propuesta de cambiar el diseño del proyecto a uno basado en el videojuego de Portal, con el motivo de hacer más atrayente este proyecto a los empleados de la empresa,

igualándolo al aspecto de un videojuego popular que ellos podrían identificar fácilmente. Este diseño es similar a un proyecto realizado en Atlassian, el cual tenía el mismo propósito de lograr telepresencia a través de una videollamada.

Figura 5.2.2-2 Propuesta de portal para el proyecto

El proyecto cambió de nombre a PortalNS. Se definió como un proyecto que establece comunicación en tiempo real entre las tres oficinas de Nearsoft, que están localizadas en Hermosillo, Chihuahua y la Ciudad de México. A través de un sistema audiovisual construido como un portal de ciencia ficción, ofrece una interfaz realista y una experiencia natural para los usuarios.

Características de PortalNS

Las principales características de PortalNS son:

- Diseño futurístico que se combina con el ambiente de las oficinas
- Interruptor de apagado/encendido
- Comunicación de voz y video ininterrumpida
- Comunicación segura y encriptada

Arquitectura y diseño

El diseño del PortalNS es una de las principales características, este sistema fue hecho para parecerse a un portal de ciencia ficción para proveer una experiencia futurística pero realista al usarlo.

Este sistema utiliza una aplicación webRTC junto con Node.js, Socket.io y Express.js.

Se utilizaron computadoras Mac Mini debido a que ofrecen un procesamiento de video suficiente para ofrecer una experiencia sin percances.

El sistema usa dos cámaras web de 1080p para que la imagen de las personas se vea con la mayor claridad y nitidez posible. También se utilizaron computadoras Raspberry Pi v3.0 en las cuales se ejecuta un servidor que recibe la señal de apagado y encendido de dos arduinos para reanudar o pausar la comunicación.

Figura 5.2.2-3 PortalNS en funcionamiento

Al finalizar esta etapa, el proyecto fue presentado a los encargados del Academy y a los fundadores de la empresa, concluyendo con las experiencias y aprendizajes de cada uno de los integrantes del Academy.

5.3 “Open Source” Phase

Open source es el nombre de la tercera etapa, al igual que las anteriores tiene una duración de 1 mes. El propósito principal de esta etapa es que el interno logre ampliar su currículum y experiencia colaborando con proyectos externos.

En esta fase se escogieron dos proyectos de código libre ubicados en la plataforma GitHub. Los proyectos escogidos fueron Materialize y Spark.

Las colaboraciones en estos proyectos incluyeron:

* resolver dudas de la comunidad de desarrolladores sobre el uso o comportamiento del proyecto.

* Aportar código para nuevas funcionalidades.

* Resolver errores del proyecto.

5.3.1 Materialize

Materialize es un framework utilizado para construir páginas web con estilos responsivos para el lado cliente inspirado en el diseño llamado “Material Design” creado por Google. Actualmente utilizado en dispositivos móviles con sistema operativo android.

Figura 5.3.1-1 Logo de Materialize

El practicante logró realizar un commit en el proyecto de código abierto, en el que se resolvía uno de los bugs reportados en el repositorio de github. Este problema consistía en que el elemento de slider (galería de imágenes) no funcionaba de manera correcta al contar con una solo imagen y, al momento de recorrer a la izquierda o derecha, la imagen desaparecía por completo. La solución consistió en añadir una condicional que checara que el número de slides fuera menor o igual a uno antes de volver a la posición inicial. Los cambios se detallan en la figura 6.1-2.

Los cambios fueron aceptados por el dueño del repositorio y se encuentra mergeado, es decir, este cambio está en el código del proyecto.

Por otra parte, el practicante también participó dentro del repositorio de Materialize participando en las discusiones dentro del repositorio, ofreciendo propuestas para la resolución de problemas de otros usuarios del framework.

Figura 5.3.1-2 Solución al bug del elemento slider de Materialize

5.3.2 Spark

Spark es un micro framework utilizado para crear páginas web con uso del lenguaje Java de manera sencilla. Este framework es bastante útil para crear micro servicios y web API's.

En este repositorio, el practicante trató de solucionar un problema reportado en el que el servicio REST no mapeaba bien las URI si estas terminaban con el símbolo de diagonal ("/"). Se realizó un pull request que consistía en añadir una condicional que checara si una uri terminaba en diagonal. Si esta condicional se cumplía, se removía de la URI para que el request se redireccionara al endpoint correcto.

Sin embargo, este pull request no fue atendido durante el tiempo de esta fase. Finalmente, este commit fue rechazado, ya que el dueño del repositorio asegura que este modificación al código cambiaría tanto el comportamiento del framework, lo cual significaría que sería necesario realizar una nueva versión del framework.

5.3 “Get a Job” Phase

La última fase del programa se llama “Get a job” la cual tiene como fin que los miembros “internos” del programa se preparen para conseguir una de las vacantes abiertas dentro de la empresa.

Durante esta etapa se consultan materiales de estudio que la empresa provee en una gran variedad de formatos como videos, cursos interactivos y lecturas. Además, en esta etapa se realizan entrevistas internas en la empresa con la intención de proveer retroalimentación de los puntos que cada interno debe reforzar para poder calificar para una entrevista con un cliente real.

Los materiales que se consultaron en esta fase incluyeron temas que todos los integrantes del Academy tenían que estudiar, entre ellos se encuentran:

- Estructura de datos
- Colecciones
- Programación orientada a objetos
- Pruebas unitarias y de integración

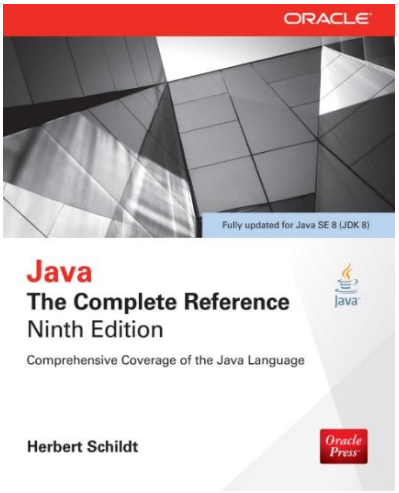

Figura 5.3-1 Portada del libro de Java The Complete Reference

Además de estos temas generales, cada participante tiene que elegir una tecnología en la cual especializarse. En esta ocasión, el practicante eligió **Java** como su tecnología principal. Para este lenguaje se realizaron otros cursos específicos en los temas de Java y sus librerías. Principalmente se enfocó en la lectura del libro *Java: The Complete Reference* de Heribert Schildt.

De acuerdo con la recomendación de su mentor, el practicante se centró en la primera parte del libro, el cual describe las características principales del lenguaje. En él se tratan temas como los tipos de datos, variables, arreglos, operadores, clases, herencia, interfaces, multihilos y métodos. Además, se incluyen características introducidas en Java 8 como los genéricos y las expresiones lambda. Otra parte importante de este libro son los capítulos que hablan acerca de la librería Collection de Java, de las cuáles es muy común escuchar en las entrevistas técnicas a los candidatos a una posición de desarrollador en Java.

Los avances en las lecturas y los cursos se documentaron en una hoja de cálculo online para monitorear el avance de cada uno de los participantes.

Al acercarse el final de esta etapa, el practicante tuvo entrevistas técnicas con varios empleados de la empresa, quienes se dedicaban en parte a desarrollar sistemas web en Java. Al finalizar las entrevistas, se hicieron las observaciones correspondientes al practicante, incluyendo puntos fuertes y débiles, además de aconsejar cómo mejorar en las entrevistas y qué material hacía falta repasar.

Terminada esta fase, se dio por concluida la generación 2016-1 de Nearsoft Academy.

6. Análisis de experiencia adquirida y resultados

En este apartado se procura hacer un análisis de las actividades realizadas durante la estancia del practicante dentro del programa de Nearsoft Academy, para lo cual se analizará cada una de las fases que se cursaron durante este período.

6.1 Primera fase

La fase de reset fue una introducción interesante al mundo laboral, más específicamente a la mentalidad con la que trabajan las empresas de desarrollo de software. En esta etapa el practicante dio cuenta de mucho de los temas que eran ajenos a él y supo la importancia de tenerlos en consideración para que, de alguna manera, la mentalidad de los nuevos desarrolladores sea más amplia. También se creó una concientización de cómo es trabajar en esta empresa en específico y las medidas necesarias al tomar al momento de trabajar con los compañeros de la empresa.

Es importante destacar que en la fase de reset te das cuenta de que el proceso de aprendizaje nunca se acaba, siempre hay más recursos, nuevas técnicas y metodologías. Es responsabilidad de los desarrolladores siempre mantenerse a la par de estos avances para no quedar rezagado.

6.2 Segunda fase

En la fase de equipo el practicante aprendió el valor del trabajo en equipo con los compañeros practicantes. Dentro del mundo laboral no hay lugar para actuar de manera egoísta o solitaria. Especialmente en el campo de la tecnología, esto se ve reflejado en los proyectos a gran escala como Google, en el que el trabajo en equipo de sus colaboradores a logrado influir en la vida diaria de todos nosotros.

Nearsoft no es la excepción. El día a día en esta compañía precisa del trabajo en equipo más que de cualquier otra cosa. Sin embargo, en esta fase, el practicante notó el desinterés de algunos compañeros en el proyecto de telepresencia, lo cual entorpeció y atrasó el desarrollo del sistema y la entrega en forma del producto. En base a esta experiencia se puede notar cómo el trabajo en equipo puede hacer que un proyecto avance rápidamente, pero una mala organización hace que en este se pierda dinero y esfuerzo.

6.3 Tercera fase

La fase de código abierto dejó como aprendizaje que leer el código de otras personas a veces puede ser complicado y confuso. Por esta razón, a veces se complica realizar aportaciones a los repositorios de estos proyectos abiertos. Alguna de las enseñanzas en esta fase fueron:

- No es necesario conocer todo el sistema para solucionar un error, solo lo necesario para entender el flujo de actividades de la parte involucrada.
- Es de vital importancia la comunicación con personas involucradas en el proyecto, tanto colaboradores como aquellos que reportan un error del sistema para darle seguimiento.
- No hace falta escribir muchas líneas de código para hacer una aportación a proyectos open source. Incluso la modificación de una sola línea puede solucionar un problema reportado.
- Es recomendable trabajar en proyectos que sean activos para poder recibir retroalimentación en el menor tiempo posible.
- Aportar en un proyecto de código abierto te ayuda a practicar el lenguaje y añade valor a tu currículum.

6.4 Cuarta fase

La última etapa de Nearsoft Academy es algo dura, pero te da las herramientas necesarias para poder realizar una entrevista técnica en cualquier empresa.

Aparte de aprender acerca de conceptos técnicos y características del lenguaje, también se aprende a cómo comportarse durante una entrevista. Al tener varios entrevistadores, el practicante adquirió conocimientos acerca de cómo pueden llegar a ser entrevistas dirigidas a desarrolladores con más experiencia. Hay que tomar en cuenta que no todas las entrevistas son iguales y depende del entrevistador y de las respuestas del entrevistado la dificultad de las preguntas.

Es muy importante aprender que no se debe improvisar ni inventar respuestas, al contrario, el entrevistado debe de reconocer cuando no se sabe la respuesta a una de las preguntas.

Al finalizar esta fase, el practicante fue contratado como desarrollador en Nearsoft, donde participó en varios proyectos internos.

7. Conclusiones y recomendaciones

Las prácticas profesionales se realizaron en una empresa de alto reconocimiento a nivel nacional. El programa de Nearsoft Academy ofrece herramientas valiosas para que sus practicantes puedan hacer de frente a las oportunidades laborales dentro de la industria de la tecnología.

A pesar de no ser una institución educativa, el practicante aprendió como trabajar con nuevos lenguajes, metodologías y herramientas que no había utilizado anteriormente. Además del aspecto técnico, el practicante aprendió nuevas técnicas de trabajo en equipo, necesarias en cualquier empresa para la realización de sus objetivos y tareas.

El practicante recomienda empezar a desarrollar habilidades técnicas durante el transcurso de su carrera, ya sea aprender un nuevo lenguaje o realizar aportaciones a proyectos de código abierto. Esto ayudará mucho a construir la experiencia de los alumnos y formará parte importante de su currículum, ya que de esta manera habrá tenido aportaciones a proyectos utilizados en empresas de todo el mundo.

Índice de figuras

Referencias

[1] G. N. R. III, "Las diferencias entre las organizaciones horizontal y vertical," [Online]. Available: https://pyme.lavoztx.com/las-diferencias-entre-las-organizaciones-horizontal-y-vertical-5849.html.

[2] E. Martínez, «Las 8 grandes ventajas de las metodologías ágiles,» IEBS, 7 Febrero 2014. [En línea]. Available: https://www.iebschool.com/blog/que-es-agile-agile-scrum/.

[3] M. Perez, «The Nearsoft Academy,» Nearsoft, 7 Mayo 2014. [En línea]. Available: https://nearsoft.com/blog/the-nearsoft-academy/.

[4] M. rubio, «Cómo funciona el sistema web,» Altenwald, 16 Noviembre 2010. [En línea]. Available: http://altenwald.org/2010/11/16/como-funciona-el-sistema-web/.

[5] O. Pastrana, «5 beneficios de aplicar metodologías ágiles en el desarrollo de software,» Intelligence to Business, 2015. [En línea]. Available: http://www.i2btech.com/blog-i2b/tech-deployment/5-beneficios-de-aplicar-metodologias-agiles-en-el-desarrollo-de-software/.

[6] «Programación Orientada a Objetos,» UM, [En línea]. Available: http://www.um.es/docencia/barzana/DIVULGACION/INFORMATICA/Prog_Obj01.html.

[7] E. Díaz, «¿Qué es la programación funcional?,» 11 noviembre 2012. [En línea]. Available: http://www.programando.org/blog/2012/11/que-es-la-programacion-funcional/.

[8] Oracle, «¿Qué es la tecnología Java y para qué la necesito?,» Oracle Corporation, [En línea]. Available: https://www.java.com/es/download/faq/whatis_java.xml.

[9] Ictea, «¿Qué es el lenguaje de programación JAVA?,» ICTEA, [En línea]. Available: http://cs.ictea.com/knowledgebase.php?action=displayarticle&id=8790.

[10] U. d. Valencia, «Javascript y Java,» Universitat de València, [En línea]. Available: https://www.uv.es/jac/guia/jscript/javascr01.htm.

[11] P. E. Goette, «Elixir,» Genbetadev, [En línea]. Available: https://www.genbetadev.com/frameworks/elixir.

[12] «Node.js: ¿Qué es y para qué sirve NodeJS?,» Netconsulting, 30 septiembre 2015. [En línea]. Available: https://www.netconsulting.es/blog/nodejs/.

[13] «Fundamentos de Git,» git-scm, [En línea]. Available: https://git-scm.com/book/es/v1/Empezando-Fundamentos-de-Git.

[14] «De la videoconferencia a la telepresencia,» Unión Internacional de Telecomunicaciones, [En línea]. Available: http://www.itu.int/itunews/manager/display.asp?lang=es&year=2008&issue=01&ipage=telepresence.

15 de Febrero del 2018. Hermosillo, Sonora.

M.C. Félix Montaño Valle
Coordinación de Prácticas profesionales de I.I.S.
Universidad de Sonora
Presente.-

Por medio de la presente, hago constar que la alumno **Paulina Alejandra Pacheco Morales**, con número de expediente **209202584** en la Carrera de **Ingeniería en Sistemas de Información** en la **Universidad de Sonora** ,ha cumplido satisfactoriamente sus prácticas profesionales en Nearsoft con un total de 400 horas en el periodo de Marzo a Junio del 2016, en el programa Nearsoft Academy, Iniciativa la cual ha sido creada para apoyar a los estudiantes universitarios a crecer en sus habilidades mediante un entrenamiento de 4 a 6 meses.

Agradezco su atención.

Atentamente.

C CUBE TECHNOLOGIES Y ASOCIADOS, S.C.
Blvd. Antonio Quiroga No. 21
Col. El Llano, Hermosillo, Sonora.
C.P. 83210
R.F.C. CCT-050718-EC1

Lic. Christian Bojorquez Villarreal
Vinculación

near soft

FPP-3

DIVISIÓN DE INGENIERÍA

COORDINACIÓN DE PRÁCTICAS PROFESIONALES
DEPARTAMENTO DE INGENIERÍA INDUSTRIAL

LIBERACION DE PRACTICA PROFESIONAL
Para acreditación de ESTANCIA PROFESIONAL

Hermosillo, Sonora, 21 / Febrero / 2018

En mi carácter de Tutor de Prácticas Profesionales, hago constar que:

I. El alumno(a) Paulina Alejandra Pacheco Morales del Programa de Ingeniería en Sistemas de Información con expediente 209202584 ha cumplido formalmente en tiempo y forma con la entrega oportuna y profesional de:
   - *Los reportes de avances periódicos de su Práctica Profesional (FPP-2)*
   - *El informe técnico del proyecto realizado.*
   - *La carta de agradecimiento a la empresa por permitir desarrollar sus prácticas profesionales*
   - *La carta formal por parte de la empresa donde hace constar el total de horas y periodo de la estancia profesional del alumno(a).*

II. He corroborado que los contenidos y tiempos de los reportes de avances están acordes con lo planeado en los anexos del formato de inscripción FPP-1 y que los contenidos y forma del *informe técnico* satisfacen los requerimientos especificados en la normatividad.

III. El número de horas acumuladas de práctica profesional, de acuerdo a los reportes de avance, es de con numero 400 con letra cuatrocientas horas

Por lo anteriormente expuesto, no tengo inconveniente alguno en dar por liberado(a), al (la) alumno(a), anteriormente referido(a), del cumplimiento de la práctica profesional para la acreditación de la ESTANCIA PROFESIONAL de manera:

☒ **TOTAL** y evaluarlo(a) con 20 créditos cumplidos.

Debido a que el alumno no terminó su práctica profesional en la empresa asignada, en base en sus reportes de avances, y dado que no ha acumulado _____ horas de práctica como mínimo, no tengo inconveniente alguno en dar por liberado(a), al (la) alumno(a), anteriormente referido(a), del cumplimiento de la práctica profesional para la acreditación de la ESTANCIA PROFESIONAL de manera:

☐ **PARCIAL** y evaluarlo(a) con los siguientes créditos, con número _____ con letra _____.

Razones generales por no haber terminado la Práctica Profesional: _____

Federico M. Ciretti Galan	Mario Barcelo	Israel Miranda Pasos
NOMBRE Y FIRMA DEL TUTOR DE PRÁCTICAS PROFESIONALES	**NOMBRE Y FIRMA DE COORDINACIÓN/RESPONSABLE DE PRACTICAS PROFESIONALES DEL PROGRAMA**	**NOMBRE Y FIRMA DE COORDINACIÓN DIVISIONAL DE PRÁCTICAS PROFESIONALES**

Original: Coordinación/Responsable de Prácticas Profesionales
Copias: 1) Tutor de Prácticas Profesionales, 2) Alumno