Informe final de practica empresarial en la empresa Innovati SAS

Roy Kleyder Maestre Ruda

Informe de práctica para optar al título de
Ingeniero de Sistemas

Tutor
Diana Margot Lopez Herrera
Ingeniería de sistemas.

Universidad de Antioquia
Facultad de Ingeniería
Ingeniería de sistemas
Medellín
2024

Cita	Maestre Ruda [1]
Referencia Estilo IEEE (2020)	[1] Maestre Ruda, “Informe final de práctica empresarial Innovati SAS”, Semestre de industria, Ingeniería de sistemas, Universidad de Antioquia, Medellín, 2024.

creative commons CC BY NC SA

UNIVERSIDAD DE ANTIOQUIA
Vicerrectoría de Docencia

Sistema de Bibliotecas

Centro de documentación de ingeniería (CENDOI)

Repositorio Institucional: http://bibliotecadigital.udea.edu.co

Universidad de Antioquia - www.udea.edu.co

El contenido de esta obra corresponde al derecho de expresión de los autores y no compromete el pensamiento institucional de la Universidad de Antioquia ni desata su responsabilidad frente a terceros. Los autores asumen la responsabilidad por los derechos de autor y conexos.

Dedicatoria

A mi papá y a mi mamá que siempre han sido un apoyo, me han brindado oportunidades y mucha paciencia. Gracias.

Agradecimientos

A Diana Margot Lopez.
A el semillero de videojuegos de la universidad de antioquia.
A mis amigos y compañeros de la carrera.
A Pablo Jaramillo y al equipo de Innovati S.A. que hicieron esta experiencia posible.

Tabla de contenido

LISTA DE FIGURAS

RESUMEN

Este documento reseña el trabajo realizado durante el proceso de prácticas profesionales en la empresa de desarrollo de software e innovación Innovati SAS. El trabajo realizado se centra en el desarrollo de la aplicación de Nati, un producto ofrecido como servició de asistente virtual en línea para empresas. Nati es un ecosistema de aplicaciones orientadas a usar las ventajas del procesamiento de lenguaje natural para ofrecer servicios de atención al cliente, asesoramiento de productos y más.

Durante el tiempo de prácticas se abordó el proyecto en dos flancos, el principal como desarrollador Unity, garantizando el correcto funcionamiento de la aplicación encargada de producir los gráficos 3D del avatar Nati, sus animaciones y el audio correspondiente. El segundo como desarrollador backend en la API de Transversales, donde se llevan a cabo todas las operaciones de persistencia de la aplicación de Nati y su ecosistema. Siguiendo los lineamientos del framework de desarrollo Scrum, los resultados obtenidos permitieron transformar Nati en una aplicación multitenant, capaz de adaptarse a diferentes requerimientos y operar para distintos clientes.

Palabras clave **— Persona Digital, Unity, Webgl, Addresables, Avatar, Lenguaje natural.**

ABSTRACT

This document reviews the work carried out during the professional internship at the software development and innovation company Innovati SAS. The work focused on the development of the Nati application, a product offered as an online virtual assistant service for businesses. Nati is an ecosystem of applications aimed at leveraging the advantages of natural language processing to provide customer service, product advice, and more.

During the internship period, the project was approached from two fronts. Primarily, as a Unity developer, ensuring the proper functioning of the application responsible for producing the 3D graphics of the Nati avatar, its animations, and the corresponding audio. Secondly, as a backend developer in the Transversales API, where all persistence operations of the Nati application and its ecosystem are carried out. Following the guidelines of the Scrum development framework, the results obtained allowed Nati to be transformed into a multi-tenant application, capable of adapting to different requirements and operating for various clients.

Keywords **—Unity, Digital Person, Webgl, Addresables, Avatar, Natural Language**

I. INTRODUCCIÓN

La práctica profesional, se realizó en la empresa Innovati SAS. Se ingresa como desarrollador Unity y backend, trabajando para completar el objetivo actual de la empresa con el producto de software Nati.

La “Plataforma de Personas Digitales Nati” es una solución tecnológica que automatiza los procesos de atención al cliente mediante el uso de herramientas de procesamiento de lenguaje natural y de un avatar que representa a una persona en un entorno 3D. Nati busca brindar asistencia y recopilar información de los usuarios según sus necesidades. Gracias a esto, la plataforma es capaz de llevar a cabo entrevistas, proporcionar asesoramiento sobre productos y resolver solicitudes y trámites de manera eficiente sin necesitar intervención humana.

Nati existe como un conjunto de aplicaciones web y backend que se comunican para conseguir realizar todas las tareas y flujos de manera flexible y modular. Nati se presenta como una plataforma de software como servicio que cuenta con una estructura flexible que permite adaptarse a las necesidades de diferentes clientes mediante una estructura multitenant. Nati cuenta también con un panel de administración que se presenta como una plataforma low-code en el que los clientes pueden configurar el uso de Nati para su empresa y crear la solución a sus necesidades.

Durante la práctica profesional se toman como principales objetivos el desarrollo del proyecto de Unity que se encarga del apartado 3D del personaje, la migración de la mayoría de sus características a un proyecto web externo a unity y el acompañamiento en el desarrollo de las funcionalidades backend de la plataforma transversales.

A continuación se presentan detalles técnicos y del proceso de práctica importantes y ejemplificantes para los futuros practicantes.

II. PLANTEAMIENTO DEL PROBLEMA

Innovati SAS requiere un practicante que pueda realizar las tareas de desarrollador de software con conocimientos en Unity, específicamente en la presentación de un avatar en un entorno 3D usando la plataforma para navegadores WebGL.

La aplicación de Nati está construida como un software a la medida para un cliente en específico. Sin embargo, hay un gran potencial al poder usar la tecnología desarrollada con diferentes clientes, adaptándola a las necesidades de cada una. Para lograr dicho objetivo, la empresa se plantea una serie de objetivos que puede ser resumidos de la siguiente manera:

- Cambio de plataforma, de windows a web. De esta manera la aplicación puede ejecutarse desde cualquier navegador, alcanzando un mayor número de usuarios.
- Cambios en su configuración. Haciendo que la aplicación funcione como una plataforma multitenant, se podría configurar fácilmente su operación para adaptarse a las necesidades de diferentes clientes.
- Introducción de un panel de administración. Donde cada uno de los clientes podría configurar la manera en la que quiere que Nati funcione de cara a sus usuarios.

De estos tres objetivos, los dos primeros requieren una fuerte participación del desarrollador encargado del proyecto de Unity. Por lo que los objetivos de la práctica se alinean a estos objetivos.

Para hacer de Nati una plataforma multitenant, una de las necesidades es poder cambiar el avatar dependiendo de los requerimientos del cliente. Por lo que poder descargar el avatar indicado al momento de ejecución es uno de los objetivos fundamentales de la práctica.

III. JUSTIFICACIÓN

Como miembro del semillero de videojuegos de la Universidad de Antioquia, que se presente la oportunidad de trabajar con la herramiento de desarrollo de videojuegos Unity durante el proceso de prácticas empresariales, representa una oportunidad de poner a prueba todos los conocimientos adquiridos en el semillero y mejorar las capacidades de manejo de la herramienta en un entorno externo al desarrollo de videojuegos.

El proyecto requiere no solamente de un ingeniero en formación con conocimientos en Unity y desarrollo web, sino de un proceso de transformación de código ya existente a una tecnología superior y un proceso de innovación que hace que la práctica y los restos que se presentan sean viables como práctica profesional en Ingeniería.

Uno de los retos de esta práctica es garantizar el funcionamiento de la aplicación en una plataforma web, dado que la implementación web de Unity presenta notables deficiencias, el cambio de plataforma de windows a web es un objetivo para nada trivial. La optimización del proyecto de Unity y la distribución de responsabilidades entre dicho proyecto y el proyecto web que lo aloja se convierte en un objetivo clave de la práctica.

Adicionalmente, Innovati requiere un desarrollador flexible que tenga capacidad de adquirir nuevos conocimientos que sean requeridos para suplir las necesidades cambiantes de una empresa que está enfocada en la innovación. Estas otras posibles responsabilidades implican la participación en el desarrollo de aplicaciones adicionales pero integradas a todo el sistema de Nati, el cual presenta también componentes de frontend web y backend.

IV. OBJETIVOS

A. Objetivo general

Actualizar el producto software Nati a una plataforma multitenant mediante la investigación, e implementación de soluciones a los diferentes retos que esto implica, conservando tecnologías base Unity y su backend web.

B. Objetivos específicos

- Comprender la arquitectura hexagonal de Transversales y la arquitectura limpia de Nati-Unity para apoyar en el desarrollo de las funcionalidades faltantes.
- Desarrollar distribución de contenidos desde la nube para la carga de avatares según pedido del tenant.
- Migrar el funcionamiento de la interfaz de usuario de Nati fuera de Unity hacia el proyecto de web de frontend.
- Construir funcionalidades extra para la API de Transversales y las necesidades del panel de administración.
- Automatizar el proceso de generación de nuevos avatares a pedido del cliente en el proyecto frontend en Unity.

V. MARCO TEÓRICO

En este marco teórico se presentan herramientas, tecnologías y conceptos necesarios para comprender lo presentado en este documento:

Persona Digital: Acorde a lo planteado por Innovati. Una persona digital es un avatar humano capaz de comunicar emociones usando su cara y cuerpo, lo cual le ayuda a establecer una conexión más cercana con su interlocutor [1].

Procesamiento de lenguaje natural: NLP (de sus siglas en inglés Natural Language Processing) es un campo de la inteligencia artificial y la informática que se centra en la interacción entre las computadoras y el lenguaje humano. Su objetivo principal es permitir que las máquinas comprendan, interpreten y generen texto y discurso de manera similar a como lo haría un ser humano. El NLP implica tareas como el análisis de sentimiento, la traducción automática, la extracción de información, la generación de texto y la comprensión del lenguaje natural en todas sus complejidades, incluyendo la sintaxis, la semántica y la pragmática. El procesamiento de lenguaje natural se utiliza en una amplia gama de aplicaciones, desde chatbots y asistentes virtuales hasta análisis de texto en redes sociales, resúmenes automáticos de contenido y más, lo que lo convierte en un campo crucial para la automatización y mejora de la interacción entre humanos y máquinas [2].

Aplicaciones Backend: En el contexto del desarrollo de software y aplicaciones web, se refiere a la parte de un sistema informático que está oculta a los usuarios finales y que se encarga de la lógica, la gestión de datos y la funcionalidad interna de la aplicación. En otras palabras, es la parte "detrás de escena" que permite que una aplicación funcione sin que los usuarios sean conscientes de ello. El backend maneja tareas como el almacenamiento de datos, la administración de bases de datos, el procesamiento de solicitudes, la gestión de usuarios y la seguridad. Por lo tanto, es esencial para el funcionamiento y la operatividad de una aplicación, ya que interactúa con el frontend (la parte visible para los usuarios) para proporcionar una experiencia completa y funcional [3].

API: Una Interfaz de Programación de Aplicaciones (API, por sus siglas en inglés, Application Programming Interface) es un conjunto de reglas y protocolos que permiten a

diferentes componentes de software comunicarse e interactuar entre sí. En esencia, una API define los métodos y las estructuras de datos que los desarrolladores pueden utilizar para acceder y manipular las funciones de un sistema o servicio de software de manera programática, sin necesidad de comprender todos los detalles internos del sistema [4].

Multitenant: Se refiere a una arquitectura o modelo en el que un solo sistema o aplicación sirve y soporta múltiples "inquilinos" o usuarios de manera simultánea, por lo general, de forma aislada y segura. Cada inquilino es una entidad independiente, como una empresa, una organización o un usuario individual, que utiliza el sistema sin tener conocimiento o acceso a los datos o recursos de otros inquilinos [5].

El enfoque multitenant es común en aplicaciones SaaS (Software como Servicio), donde varios clientes comparten la misma instancia de la aplicación, pero sus datos y configuraciones se mantienen separados y seguros. Esto permite una mayor eficiencia en términos de recursos y gestión, ya que una única instancia de la aplicación puede atender a múltiples clientes. Esta es una característica importante en el diseño de sistemas escalables y rentables, particularmente en entornos de nube y servicios compartidos.

Low-Code: Se refiere a una metodología de desarrollo de software que se centra en simplificar el proceso de creación de aplicaciones al reducir la cantidad de código de programación manual necesario. En un entorno de desarrollo de "low-code", se proporcionan herramientas y plataformas que permiten a los desarrolladores y a personas con menos experiencia en programación diseñar y crear aplicaciones utilizando interfaces visuales y componentes predefinidos [6].

Algunas de las tecnologías usadas son:

Unity: es un motor de desarrollo de videojuegos ampliamente reconocido en la industria, se erige como un pilar fundamental en la creación de experiencias interactivas en 2D y 3D. Su versatilidad y robustez lo convierten en una herramienta esencial para desarrolladores y educadores en el campo de la programación y el diseño de videojuegos. Con soporte para múltiples lenguajes de programación, incluyendo C# [7]. Unity permite la creación de proyectos multiplataforma con eficiencia y precisión. Su sólida comunidad y abundantes recursos de aprendizaje lo convierten en un recurso invaluable tanto para la industria como para la educación.

Unity es una de las principales herramientas de trabajo en esta práctica. Al comenzar el proceso de práctica, Nati estaba construida principalmente en Unity. Esta tecnología se encargaba tanto de procesar los gráficos 3D como la interfaz de usuario 2D. Una de las principales debilidades de Unity es su distribución a través de la plataforma web, ejecutando sus aplicaciones en el navegador, debido a que las aplicaciones resultantes de Unity tienen una alta demanda de procesamiento gráfico, siendo este un requerimiento difícil de cumplir en toda la variedad de dispositivos con un navegador web.

Addresables Assets: Los assets en Unity son componentes externos usados en la ejecución de la aplicación. Estos deben tener ciertas configuraciones para funcionar y dichas configuraciones se suelen realizar desde el editor de Unity. Addressable Assets es una tecnología ofrecida por Unity que permite tener paquetes de assets por fuera de la versión compilada de la aplicación que puedan ser descargados y utilizados a discreción durante la ejecución de la aplicación. Esta tecnología es clave para poder tener diferentes avatares que satisfagan los tenant de Nati [8].

WebGL™: es una herramienta multiplataforma, de código abierto basada en OpenGL, cuya función es el renderizado de gráficos 3D en un navegador web [2]. Esta tecnología es la base que permite que el proyecto 3D de Unity pueda ser ejecutado en los navegadores web de computadores y teléfonos móviles [9].

.NET o Dotnet, desarrollado por Microsoft, es un entorno de desarrollo y ejecución de aplicaciones de software que se utiliza ampliamente en la creación de sistemas backend. Ofrece una amplia variedad de lenguajes de programación, como C# y Visual Basic, lo que permite a los desarrolladores diseñar aplicaciones escalables y seguras. Dotnet se basa en un enfoque de código administrado que proporciona características avanzadas para la gestión de memoria y seguridad. Además, su framework proporciona una amplia gama de bibliotecas y servicios que simplifican la creación de aplicaciones robustas y de alto rendimiento. En el contexto del desarrollo backend, .NET se destaca por su capacidad para implementar servicios web, aplicaciones empresariales y sistemas de bases de datos, lo que lo convierte en una elección destacada para proyectos que requieren un sólido soporte en el lado del servidor [10].

Angular: es un framework de desarrollo frontend web desarrollado en Typescript. Está framework permite la construcción de aplicaciones web basadas en componentes de fácil escalabilidad. Con el uso de componentes reutilizables, Angular busca ofrecer una arquitectura limpia y eficiente en sus aplicaciones [11]. En Innovati, Angular es usado en las aplicaciones de Nati, como contenedor del proyecto de Unity y la interfaz consecuente como en la aplicación del panel de administración donde se configura cada tenant y servicio.

VI. METODOLOGÍA

La metodología de trabajo durante todo el ciclo de desarrollo de la práctica se ajustó a los lineamientos del framework de desarrollo ágil Scrum con algunas modificaciones. Los diferentes eventos que componen esta metodología en el equipo se presentan de la siguiente manera.

Reuniones de fijación de objetivos a corto y mediano plazo. Estas reuniones se llevaban a cabo con el equipo y el producto owner, en ellas se establece una visión del equipo para los siguientes meses y se fijaban objetivos a cumplir. Estos objetivos se reconocían como features y se asignan a algunas de las épicas que ya existían en los planteamientos del equipo.

Luego, el product owner se encarga de priorizar las features establecidas, crear historia de usuario que permitieran al equipo enfocar su trabajo en objetivos concretos y darle un orden de importancia que seguiría el equipo durante los próximos meses.

Durante el ciclo iterativo de los sprints se tenían diferentes eventos que delimitaban el trabajo a realizar, el primero de ellos es la planeación, en esta se seleccionan las historias de usuario que se trabajan en el sprint y se creaban tareas que ayudarán a resolverla. Todo bajo una limitante de los diez días de trabajo contabilizados por horas y estimando el tiempo que tomaba cada tarea.

Así, todo el equipo de trabajo conocía sus responsabilidades y se dedicaba a trabajar durante el sprint. Tres veces por semana se realizaban reuniones de sincronización donde el equipo comentaba sus avances y las tareas que se iban a realizar en los días siguientes, citando reuniones necesarias cuando las tareas no dependen de un solo miembro del equipo o señalando las tareas ya completadas para que la persona encargada de las pruebas integrales procediera a ejecutarlas.

Para el final del sprint se realizaban reviews de los objetivos logrados por el equipo. En estas se recorren todas las tareas establecidas en un inicio y se iba mostrando, explicando cómo se lograron en caso de completarse o el porque no se pudieron terminar en caso de no completarse. Las tareas no completadas, en la mayoría de los casos se transfieren al siguiente sprint o se reevalúa su prioridad pudiendo retrasarse uno o más sprints.

El producto de software, Nati, ha pasado por varias etapas de desarrollo durante por lo menos dos años, sin embargo se ha visto bajo cambios drásticos en su manera de funcionar. El objetivo actual del proyecto se enfoca en alcanzar las funcionalidades multitenant que permitan vender el software como un servicio a diferentes clientes, eliminando la necesidad de trabajar este proyecto como un desarrollo a la medida y alcanzando así una flexibilidad suficiente para ser usado simultáneamente por varios clientes y supliendo necesidades variadas.

Una tarea habitual encontrada en muchos de los sprints comprendía el desarrollo de Spikes, estos eran franjas de tiempo donde los desarrolladores se dedicaban a investigar tecnologías y tendencias en el desarrollo actual, permitiendo recolectar conocimiento e ideas para compartir con los compañeros de trabajo y proponer ideas a realizar en las reunión de establecimiento de objetivos. Lo cual permitió llegar a realizar un producto innovador y ajustado a las tendencias tecnológicas vigentes.

VII. RESULTADOS

Por ser Nati un producto de propiedad de Innovati S.A.S algunos de los conceptos técnicos considerados secreto empresarial no se plasman aquí, y se da fé mediante las cartas de certificación de la compañía del correcto cumplimiento. A continuación, se presentan algunos aspectos relevantes de la práctica en el aspecto técnico y que puede ser revelado a la comunidad académica.

Migración de interfaz de usuario de la aplicación de Nati a proyecto web.

La interfaz de usuario de Unity presenta ciertas limitaciones, especialmente a la hora de mantener una distribución responsive para todas las resoluciones en las que se puede ejecutar la aplicación. Adicionalmente, el framework para UI de unity, UI Toolkit representa una barrera de entrada bastante alta para cualquier desarrollador que tenga que modificarla, ya que este no se asemeja mucho a como funciona el frontend en aplicaciones web, haciendo que modificar la interfaz de usuario de la aplicación deba realizarse exclusivamente por el desarrollador de Unity, siendo este el único responsable. Esto sumado a problemas de compatibilidad de algunos elementos en dispositivos móviles, hizo necesario realizar un cambio de paradigma para la interfaz de usuario.

En consecuencia el equipo llegó a la conclusión de que toda la interfaz de usuario de la aplicación debía ser migrada a la capa web, permitiendo modificarla a través de HTML y CSS. Mediante el trabajo en equipo del desarrollador frontend y el de Unity se logró realizar dicha migración. Las capacidades de comunicación entre el proyecto de Webgl ejecutado de manera embebida en el proyecto de Angular permitió que cada vez que Unity necesitara mostrar algo en la interfaz de usuario, pudiera enviarse una señal al proyecto web para que este pudiera renderizar en pantalla la información necesaria. De manera inversa, cualquier interacción del usuario con la interfaz envía una señal a Unity para que este responda en consecuencia.

En vista de los buenos resultados obtenidos en la migración de los componentes de la interfaz de usuario al proyecto frontend web, mejorando la definición de la interfaz, su responsividad y el rendimiento general de la aplicación. Durante el proceso de prácticas se fueron migrando diferentes componentes que eran responsabilidad de Unity hacia el proyecto web. La comunicación del chat, la interfaz con el backend, la reproducción de audio y gran parte del

sistema de carga de avatares se convirtieron en responsabilidad del proyecto web, haciendo a Unity responsable de la visualización del avatar 3D y sus animaciones.

Sistema de entrega de contenidos para descarga de avatares multitenant.

Nati, al ser un proyecto multitenant necesita poder cargar diferentes configuraciones dependiendo de los parámetros establecidos por los clientes de la aplicación. Una de ellas es la carga de un avatar personalizado para cada tenant. Al inicio de la práctica solo existían un avatar, Nati, esté avatar siempre se cargaba y su configuración estaba integrada a la aplicación misma. Si se quería tener otro avatar, era necesario compilar una versión diferente de la aplicación. Así pues, al cargar la app desde el navegador web, se descargaba toda la información de la aplicación, incluyendo el avatar predeterminado con sus texturas y configuraciones visuales preestablecidas.

El primer cambio realizado para resolver esta necesidad fue aprender a usar el sistema de carga de addresables de Unity. Este sistema permite que los juegos se puedan actualizar durante su ejecución solo descargando los assets necesarios, sin necesidad de que el usuario tenga que descargar una nueva versión del compilado de la aplicación. En este caso se aprovechó el pipeline de descarga de assets en runtime, permitiendo que la aplicación de Nati pudiera ser cargada inicialmente sin ningún avatar (haciendo que la descarga fuera menor) para luego, una vez la aplicación se está ejecutando, se pueda descargar el avatar necesario para dicho tenant.

Una explicación más detallada puede encontrarse en la Fig 1. En el flujo del proceso A se explica la carga de un avatar para ser usado por un tenant en específico. Una vez importado el avatar a Unity y configurado para su funcionamiento, este se exporta como un paquete de addressable assets. Como estos paquetes pueden ser un poco pesados para descargar cada que se abre la aplicación, primero debe ser comprimidos para que los usuario tengan menores tiempos de descarga y uso de datos. Una vez el archivo se encuentra comprimido, este se sube al servicio de nube contratado por la empresa y se asocia al avatar por medio de una petición al servicio de transversales. De esta manera, el avatar queda listo para ser usado por Unity cuando se abra la aplicación en el tenant indicado.

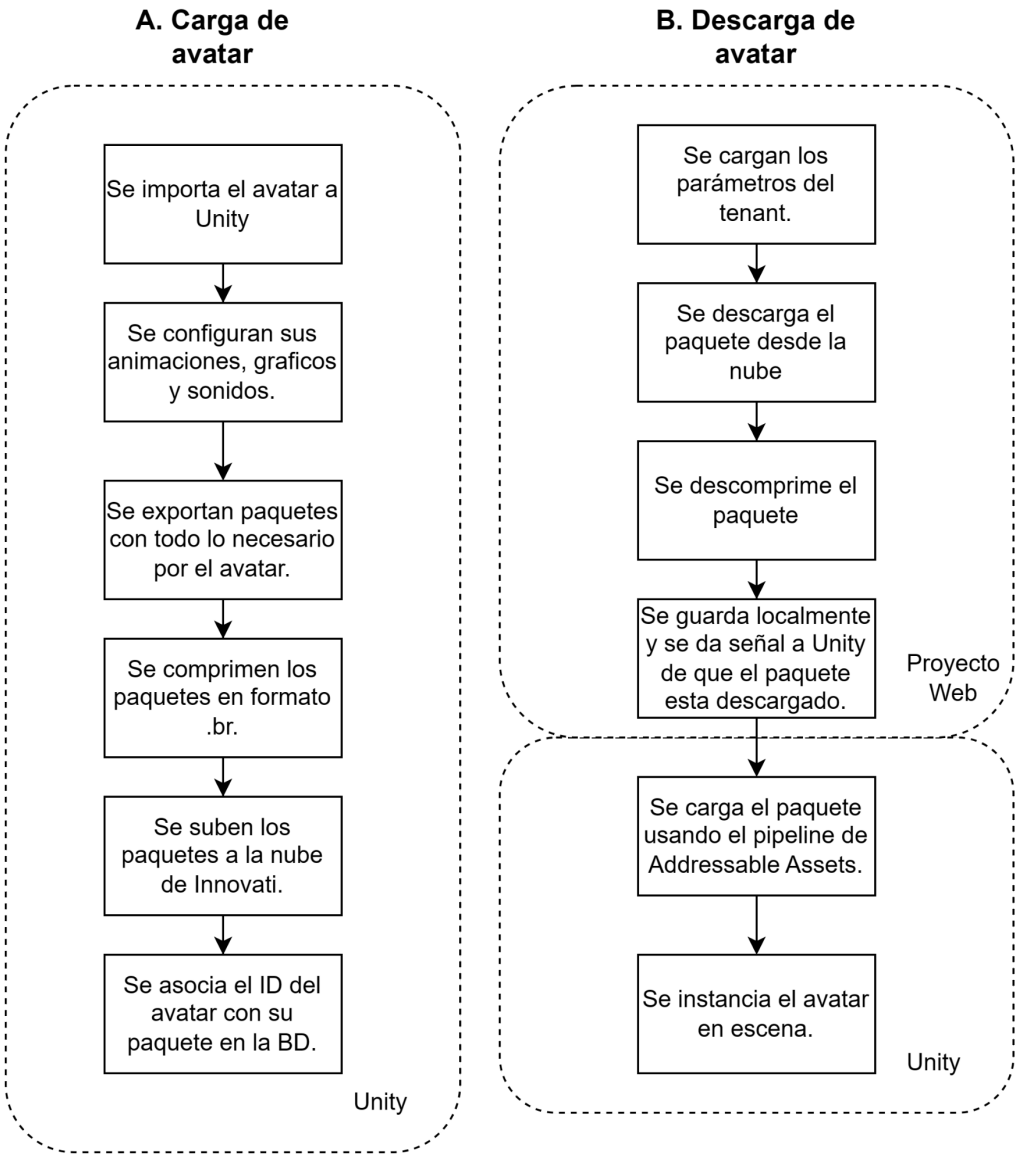

Fig. 1 A. Descripción del proceso de carga de un avatar configurado. B. Descripción del proceso de descarga de un avatar al iniciar la aplicación.

En la parte B de la Fig. 1 se observa el proceso de descarga durante la ejecución de la aplicación de Nati, en este caso, parte del proceso se realiza desde la aplicación web. Se empieza identificando cual es el avatar a descargar a partir de los parámetros de inicio de la aplicación. Después se consulta en base de datos cual es la url de descarga del avatar y se empieza su descarga. Luego se descomprime y se almacena localmente dentro del proyecto web y se envia un mensaje de aviso al proyecto de unity con la dirección del archivo. Una vez en Unity, el enlace

es asociado al addressable assets el cual se encarga de hacer la descarga dentro de la aplicación de Unity. En este caso se trata de una descarga local, no hay necesidad de acceder a la red, sino que se hace una transferencia del archivo desde el proyecto web al proyecto de Unity. Finalmente, el pipeline de addressable assets se encarga de ubicar el avatar en escena e inicializarlo.

Automatización de configuración de nuevos avatares.

Innovati ofrece una selección de avatares predeterminados de diferentes géneros y estilos que buscan ser una opción simple para que los clientes puedan configurar sus tenant de Nati. Sin embargo, para muchos clientes, tener un avatar personalizado que representé a su empresa y al mensaje que quieren dar es fundamental a la hora de implementar a Nati en sus servicios. Por lo que crear nuevos avatares, configurar sus características para que funcionen en Unity y desplegarlos en la aplicación era una tarea bastante frecuente.

Esta tarea requiere de varios pasos en los que intervienen las personas encargadas del arte y del desarrollo en Unity. De manera resumida, la implementación de un nuevo avatar implica crear el modelo en un software de modelado 3D, crear o configurar su vestimenta, exportar el modelo y las texturas a Unity. El encargado de Unity debe comprimir las texturas para disminuir su peso, configurarlas para su correcta visualización, disponer las animaciones para que se ejecuten en el momento correcto, adaptar el sistema de sincronización labial y todo el pipeline de despliegue del avatar en la app. Los cambios realizados en Unity pueden tardar hasta una hora para completarse.

Una vez desplegado el avatar, este debe recibir el visto bueno para su implementación. De manera iterativa los diferentes interesados solicitaban cambios en el avatar, que requerían que todo el proceso tuviera que repetirse una y otra vez. Como desarrollador responsable de Unity, esta implementación iterativa de los avatares requería gran parte del tiempo laboral, el cual podría aprovecharse mejor con tareas que aportaran un mayor valor a la empresa.

La automatización de este proceso hizo parte de los objetivos de las prácticas y los resultados obtenidos redujeron el tiempo invertido de manera significativa. Se construyó una ventana de configuración del avatar mediante las herramientas de editor de Unity como se observa en la Fig. 2.

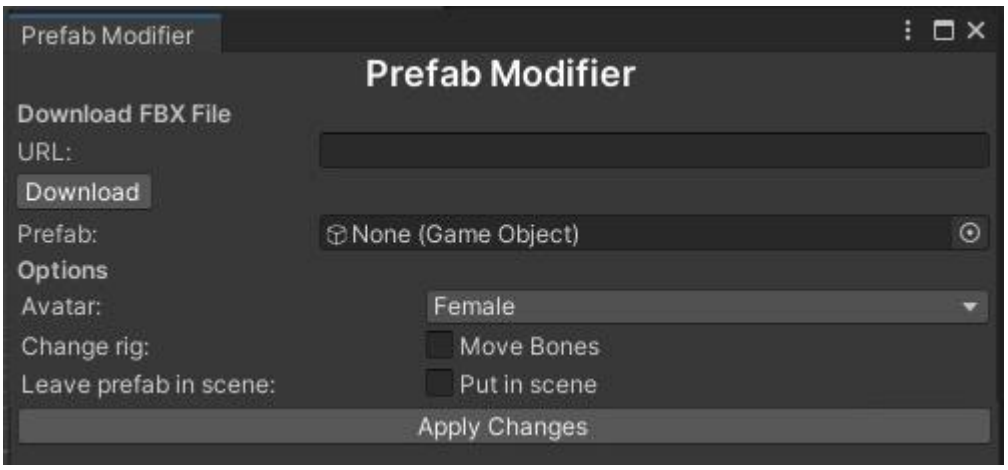

Fig. 2 Herramienta de configuración rápida de avatares.

Desde esta ventana se puede asignar un avatar o descargarlo desde una url. Una vez el avatar está establecido, es posible hacer algunas configuraciones extra relacionadas con el género del avatar y su configuración de esqueleto para las animaciones. Al dar clic en Apply changes, se obtiene un avatar completamente configurado con los componentes listos para ser usado en la aplicación.

Con una segunda ventana de configuración que se puede apreciar en la Fig 3. se puede desplegar el avatar para su carga en la aplicación de Nati y en su panel de configuración. Esta ventana se encarga de crear los paquetes de assets usados por el sistema de Addressables, comprimirlos y subirlos a el servicio de nube, para luego actualizar en base de datos su ubicación. Con estas herramientas implementadas, se intenta eliminar la intervención del desarrollador de Unity, dándole más tiempo para realizar otras tareas y permitiendo a la persona responsable del arte poder realizar todo el proceso de generación de avatares.

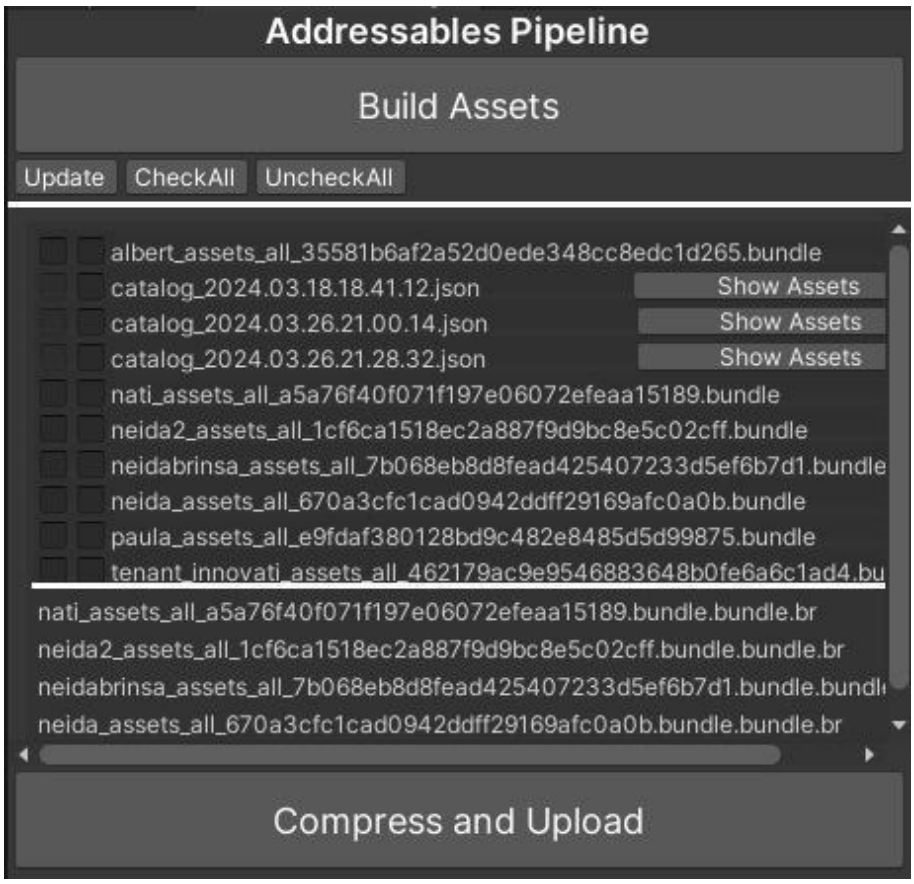

Fig. 3 Herramienta de despliegue de paquetes de addressable assets de avatares.

Trabajo en la API de transversales.

La API de transversales es una aplicación backend escrita en .Net. Esta tiene como responsabilidad principal realizar operaciones sobre la base de datos de todo el sistema de Nati. Durante el último tercio del proceso de prácticas, fue tomada la responsabilidad de satisfacer las necesidades de los nuevos requerimientos en el desarrollo de transversales. Incluyendo principalmente la generación de nuevos endpoints y actualización de los existentes antes los cambios en el modelo de negocio.

VIII. CONCLUSIONES

El proceso de prácticas profesionales se completa con éxito alcanzando los objetivos planteados en la propuesta inicial. Se alcanzó un nivel de integración con el equipo de trabajo, volviéndose una pieza fundamental en el alcance de los objetivos planteados.

Se comprendieron y manejaron las arquitecturas de los proyectos de Nati en Unity y Transversales. Esto permitió el constante aporte a los objetivos de la compañía, cubriendo las necesidades de desarrollo que surgieron durante las prácticas en cuanto al frontend 3D y el backend de transversales.

En Unity se logró alcanzar la característica multitenant del proyecto permitiendo a cada cliente configurar los diferentes avatares y sus características para ser mostrados en un versión de la aplicación. La distribución de contenidos para desplegar cada versión del avatar permitió unos tiempos de carga menores donde cada usuario solo tiene que descargar los assets necesarios para el funcionamiento del tenant de la aplicación que está usando.

La migración de la interfaz de usuario y las diferentes características que eran responsabilidad del proyecto de Unity hacia el proyecto web en Angular permitió no solo mejorar el rendimiento, el aspecto gráfico de la interfaz y su responsividad en diferentes pantallas. Sino que permitió que muchos de los cambios requeridos que anteriormente tendrias que ser cubiertos por un desarrollador Unity, pudieran ser ejecutados por otros miembros del equipo con conocimiento en desarrollo web, de esta manera se logró repartir mejor las responsabilidades y evitar cuellos de botella en el tiempo de desarrollo de las características de Nati.

Las mejoras en el pipeline de creación de nuevos avatares permitieron reducir los tiempos necesarios para desplegar las nuevas versiones de los avatares diseñados para clientes específicos, haciendo que el proceso iterativo de su creación pudiera realizarse más ágilmente. La implementación de las herramientas de configuración de avatares también hicieron posible que, miembros del equipo que no tienen tanto conocimiento en Unity pudieran realizar configuraciones y despliegues de los avatares hacia la aplicación.

Los aportes realizados en cuanto al desarrollo de la aplicación de Transversales no solo permitieron avanzar en los objetivos planteados por la empresa sino que permitieron que otros miembros del equipo pudieran desprenderse de ciertas responsabilidades, aportando una mayor flexibilidad a la hora de planear las tareas y los responsables de abordar cada una durante la planeación de los sprints.

Adicionalmente a los objetivos planteados, la inclusión de los spikes de investigación permitió no solo mantener a los desarrolladores al tanto de las tendencias tecnologías y las posibilidades que estas ofrecían, sino que también ayudaron a mantener el producto y los objetivos de la empresa a la vanguardia tecnológica. Por otro lado, este acercamiento suponia un cambio frecuente en los objetivos y las tareas a realizar por el equipo.

Por otra parte, el trabajo en equipo fue fundamental en el desarrollo de la práctica, durante esta se trabajó constantemente con los miembros del equipo encargados del desarrollo de frontend, arte y pruebas. A lo largo de la práctica en diferentes objetivos se trabajó con todos los miembros del equipo logrando una resolución de objetivos muy buena. Siempre se presentó un ambiente de hermandad en el equipo, estando siempre disponible para resolver las dudas de los otros miembros.

En contraste con el proceso académico, la metodología de trabajo se hacía bastante similar a la encontrada en asignaturas como los proyectos integradores y los diferentes componentes de la fábrica de software que se ha venido implementando en los últimos semestre en el departamento. La única gran diferencia con los procesos académicos es la velocidad en la que se ejecutan las tareas, gracias a que al ser un empleo, todos los integrantes del equipo se encuentran disponibles en horario laboral.

El constante trabajo como parte del semillero de videojuegos de la universidad de antioquia fue fundamental para la exitosa realización de la práctica. Las herramientas aprendidas en este permitieron una rápida apropiación de una de las principales herramientas de desarrollo durante el proceso.

Asimismo, también cabe resaltar que, a diferencia del desarrollo de los videojuegos del semillero, la escala del desarrollo en Unity durante la práctica se dió en una dirección diferente. Para ilustrar esta diferencia, el desarrollo en el semillero es como caminar en un gran charco que

solo te llega hasta las rodillas, donde las posibilidades son muy amplias pero de poca profundidad, donde cada desarrollo puede tener muchos componentes, pero donde el producto final se convierte en la suma de muchos sistemas de baja complejidad que trabajan en sinergia.

Por otro lado, el desarrollo usando Unity en Innovati es más como saltar a un pozo estrecho pero muy profundo, donde la investigación de herramientas muy específicas de Unity, enfocadas en su compatibilidad con la plataforma web, la optimización de su requerimientos gráficos y la comunicación con herramientas externas como el proyecto de Angular y las herramientas de la nube, se convirtieron en el foco del desarrollo. Esto ocasionalmente reveló las limitaciones de la herramienta, haciendo que se optara por construir soluciones desde cero que pudieran adaptarse al sistema existente, que continuar con las soluciones ofrecidas por Unity.

REFERENCIAS

[1] "Nati.ai," *Nati*, 2022. https://nati.ai/ (accessed Apr. 07, 2024).

[2] "¿Qué es el procesamiento de lenguaje natural? - Explicación del procesamiento de lenguaje natural - AWS," *Amazon Web Services, Inc.*, 2023. https://aws.amazon.com/es/what-is/nlp/#:~:text=tareas%20de%20NLP%3F- (accessed Apr. 12, 2024).

[3] Coursera, "What Does a Back-End Developer Do?," *Coursera*, Jun. 16, 2023. https://www.coursera.org/articles/back-end-developer

[4] "¿Qué es una API?," *www.redhat.com*, Jan. 20, 2023. https://www.redhat.com/es/topics/api/what-are-application-programming-interfaces (accessed Apr. 12, 2024).

[5] D. Alexander, "Arquitectura multitenant: qué es y por qué es importante," *Platzi*, 2017. https://platzi.com/blog/multi-tenant-que-es-y-por-que-es-importante/ (accessed Apr. 12, 2024).

[6] IBM, "What is Low-Code | IBM," *www.ibm.com*, 2023. https://www.ibm.com/topics/low-code

[7] U. Technologies, "Unity - Unity," *unity.com*. https://unity.com/es (accessed Apr. 04, 2024).

[8] "Getting started with Addressable Assets | Package Manager UI website," *Unity3d.com*, Oct. 18, 2023. https://docs.unity3d.com/Packages/com.unity.addressables@1.1/manual/AddressableAssetsGettingStarted.html (accessed Apr. 12, 2024).

[9] Khronos Group, "WebGL," *The Khronos Group*, Jul. 19, 2011. https://www.khronos.org/api/webgl

[10] Microsoft, ".NET (y .NET Core): introducción e información general," *learn.microsoft.com*, Apr. 02, 2024. https://learn.microsoft.com/es-es/dotnet/core/introduction (accessed Apr. 12, 2024).

[11] Angular, "Angular," *angular.io*, 2022. https://angular.io/guide/what-is-angular (accessed Apr. 22, 2024).