

Realidad Virtual

Proyecto final:

*“Enciclopedia visual del
cuerpo humano con
realidad aumentada”*



Alumno: Gabriel Godoy

Legajo: 10525

Tabla de contenido

Resumen.....	3
Introducción	4
¿Qué es la realidad aumentada?.....	5
¿Realidad aumentada o Realidad virtual?	7
Hardware utilizado	7
Software utilizado	8
Creación de la aplicación.....	9
Creación de la interfaz de usuario.....	10
Objetos en 3D y su modificación en Blender	12
Vuforia y Android	14
Marcadores y creación de base de datos en Vuforia	15
Incorporación de marcadores a Unity.....	16
Scripts	17
Exportación de la aplicación.....	22
Resultados	23
Conclusiones y Mejoras a futuro.....	24
Referencias:.....	25
ANEXO	26

Resumen

Para el presente informe, se muestra el desarrollo de una aplicación de realidad aumentada, la cual cuenta con la capacidad de desplegar modelos en 3D de distintas partes del cuerpo humano. Esta aplicación puede ejecutarse en cualquier dispositivo que funcione en base a un sistema operativo Android, por ejemplo en celulares Smartphone o tablets.

El objetivo principal es poder integrar la información disponible en una enciclopedia de del cuerpo humano y los beneficios que ofrece la realidad aumentada, logrando una interacción más beneficiosa a nivel visual para el usuario.

Esta aplicación está destinada a estudiantes y profesores que quieran incorporar este tipo de tecnología en sus clases durante el proceso de aprendizaje de la anatomía humana. En otras palabras este desarrollo se posiciona como una herramienta de dicho aprendizaje ya que no tan solo se basa en el despliegue de un objeto en 3D sino que además cuenta con la posibilidad de poder manipularlo así como también de mostrar en pantalla información y realzar alguna animación cargada previamente en dicho objeto.

Introducción

En la actualidad a la hora de aprender temas relacionados con la anatomía, se recurre en una primera instancia a libros y diferentes enciclopedias con ardua información. También son abundantes las enciclopedias o sitios de internet con muchos gráficos e imágenes sumamente ilustrativas que logran asemejarse a la realidad.

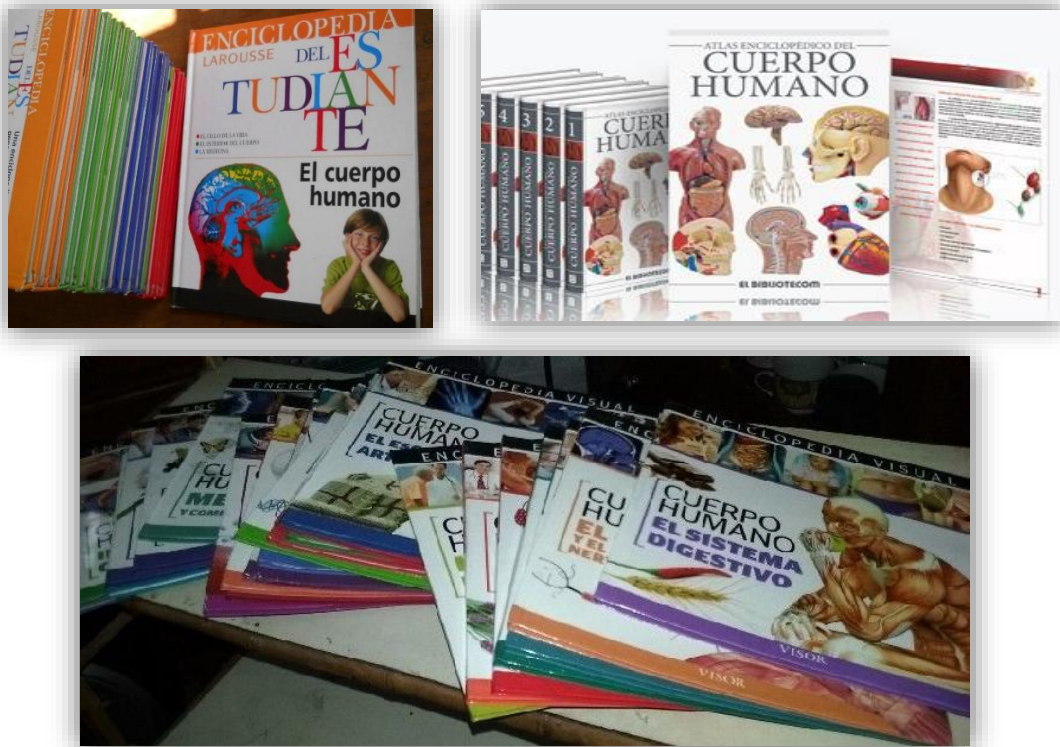


Figura 1: Enciclopedias del cuerpo humano en la actualidad

Como se puede observar este material de estudio contiene mucha información a nivel texto y visual pero no va más allá que una lectura y observación de imágenes por parte de la persona que este leyendo alguno de estos artículos.

Con el fin de incorporar una mayor interacción entre la persona y la información que se quiere exponer, se propuso agregar un plus en estas enciclopedias. La opción elegida fue el uso de la técnica en auge la actualidad, la realidad aumentada. De esta manera la persona puede aprovechar todos los beneficios que ofrecidos por esta tecnología, como poder observar un objeto 3D, manipularlo, animarlo y que reaccione ante determinadas circunstancias entre otras.

¿Qué es la realidad aumentada?

La realidad aumentada es un término que se asocia a un conjunto de tecnologías que permiten que un usuario pueda visualizar el mundo real a través de un dispositivo tecnológico (Smartphone, Tablet, PC's) con información gráfica que añade este dispositivo. Se crea de esta manera un entorno en el que la información y los objetos virtuales se fusionan con los objetos reales, ofreciendo una experiencia tal para el usuario, que puede llegar a pensar que forma parte de su realidad cotidiana, olvidando incluso la tecnología que le da soporte. En otras palabras la realidad aumentada combina elementos reales y virtuales, la interacción es en tiempo real y con componentes visualizados en 3D.

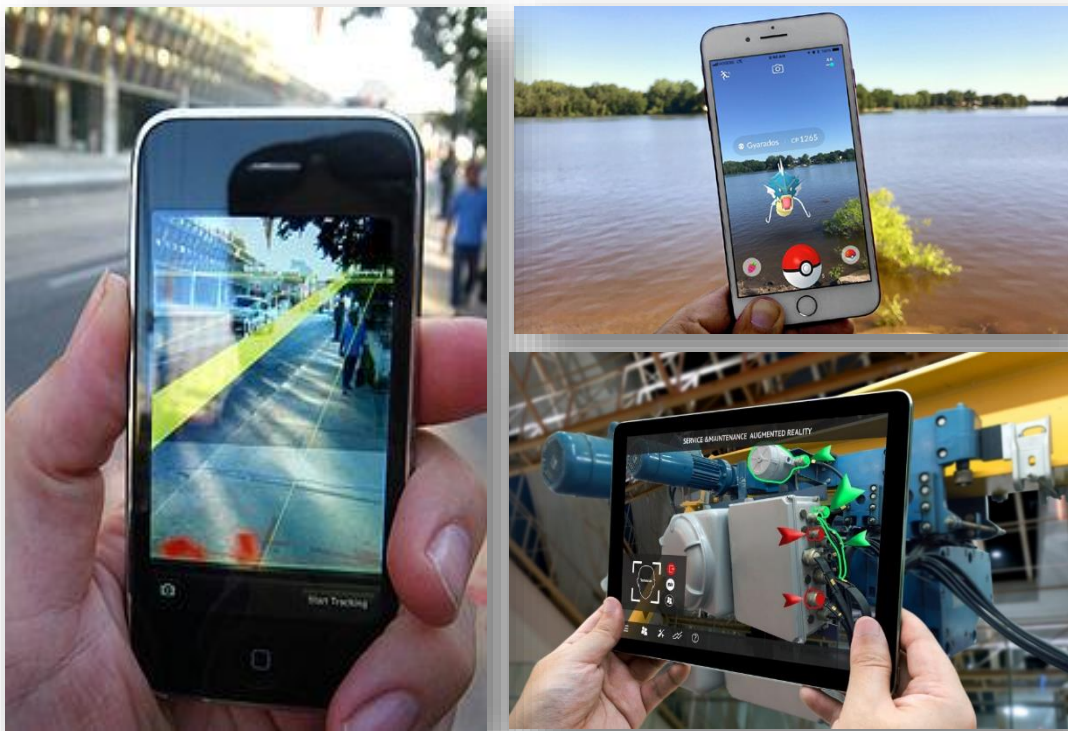


Figura 2: Ejemplos de uso de la Realidad Aumentada

De manera simplificada, por lo tanto, para componer un servicio de realidad aumentada son necesarios 4 componentes básicos:

- 1) Un elemento que capture las imágenes de la realidad que están viendo los usuarios. Por ejemplo una sencilla cámara de las que están presentes en los ordenadores o en los teléfonos móviles.



Figura 3: Camara presente en smartphones.

- 2) Un elemento sobre el que proyectar la mezcla de las imágenes reales con las imágenes sintetizadas. Para ello se puede utilizar la pantalla de un ordenador, de un teléfono móvil o de una consola de videojuegos.

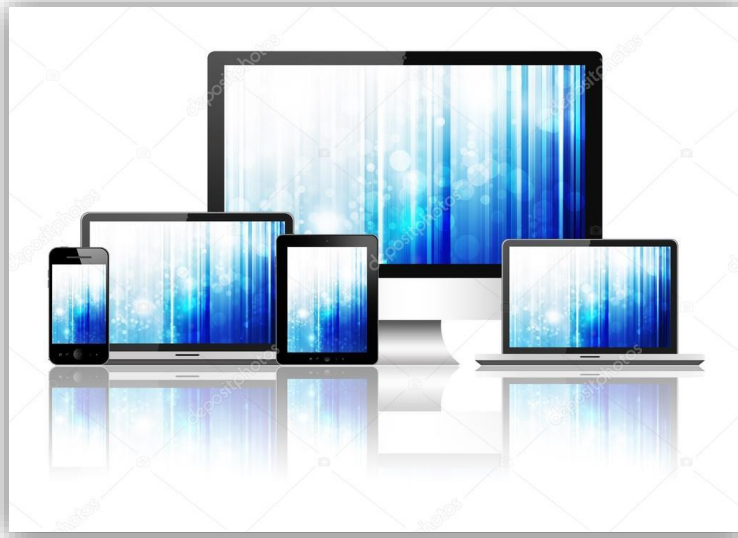


Figura 4: Pantallas de distintos dispositivos

- 3) Un elemento de procesamiento, o varios de ellos que trabajan conjuntamente. Su cometido es el de interpretar la información del mundo real que recibe el usuario, generar la información virtual que cada servicio concreto necesite y mezclarla de forma adecuada. Nuevamente encontramos en los PCs, móviles o consolas estos elementos.
- 4) Un elemento que podría denominarse activador de realidad aumentada. En un mundo ideal el activador sería la imagen que están visualizando los usuarios, ya que a partir de ella el sistema debería reaccionar. Pero, dada la complejidad técnica que este proceso requiere, en la actualidad se utilizan otros elementos que los sustituyen. Se trata entonces de elementos de localización como los GPS que en la actualidad van integrados en gran parte de los smartphones, así como las brújulas y acelerómetros que permiten identificar la posición y orientación de dichos dispositivos, así como las etiquetas o marcadores del tipo RFID o códigos bidimensionales, o en general cualquier otro elemento que sea capaz de suministrar una información equivalente a la que proporcionaría lo que ve el usuario, como por ejemplo sensores



Figura 5: Tipos de "activadores" para Realidad aumentada (marcadores y gps).

¿Realidad aumentada o Realidad virtual?

Aunque aparentemente son similares, la realidad aumentada y la realidad virtual son tecnologías muy distintas. La realidad virtual es una tecnología informática que replica artificialmente un entorno real (o imaginario) y le proporciona al usuario (apelando principalmente a su visión y audición) la sensación de estar verdaderamente dentro de él. La realidad mixta abarcaría todo el espectro, incluyendo tanto la realidad aumentada como la realidad virtual.



Figura 6: Realidad mixta

Hardware utilizado

Para el desarrollo de la aplicación de realidad aumentada se utilizó un ordenador tipo notebook con las siguientes características:

Notebook Dell Inspiron 3421

Características generales:

Disco rígido: **500 Gb.**

Memoria RAM: 4 GB DDR3 800 MHz

Pantalla: LED 14"

Motherboard:

Marca: **Dell Inc.**

Modelo **05HG8X.**

CPU:

Marca y modelo: **Intel Core i3 3217U**

Frecuencia de trabajo: **1.8 GHz**

Potencia de cálculo de CPU: **55,073 GFLOPS**

Cantidad de núcleos: **4 CPUs.**

Placa aceleradora de gráficos:

Marca y Modelo: **Intel(R) HD Graphics 4000**

Frecuencia de trabajo: **349 MHz**



Figura 7: Computadora utilizada

Software utilizado

Para la creación de la aplicación se optó por utilizar el programa conocido como **Unity 3D**, específicamente la versión 2018.2.13. Este software es un motor de videojuego multiplataforma creado por Unity Technologies. Unity está disponible como plataforma de desarrollo para Microsoft Windows, OS X, Linux. La plataforma de desarrollo tiene soporte de compilación con diferentes tipos de plataformas. Unity tiene dos versiones: Unity Professional (pro) y Unity Personal. Unity es un programa sumamente versátil, poderoso y relativamente fácil de utilizar. La programación se realiza en lenguaje C#.

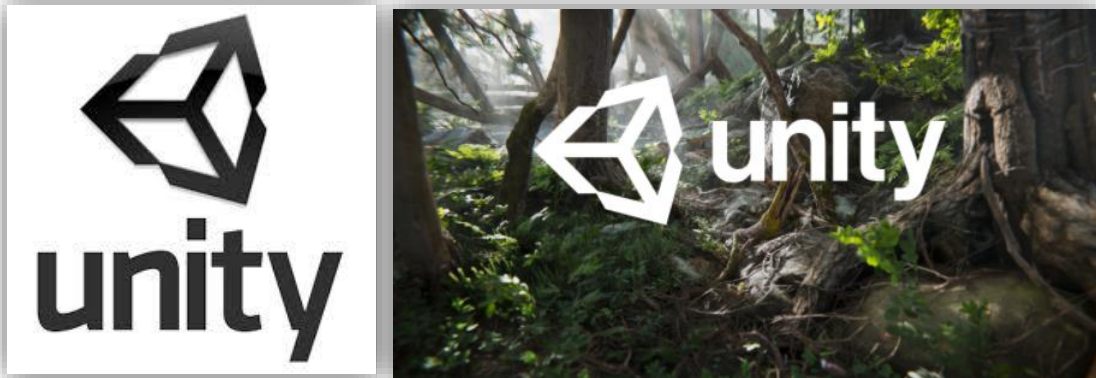


Figura 8: Logo de la plataforma Unity

Con respecto al lenguaje de programación utilizado por este software, **C#** (pronunciado si sharp en inglés) es un lenguaje de programación orientado a objetos desarrollado y estandarizado por Microsoft como parte de su plataforma .NET, que después fue aprobado como un estándar por la ECMA (ECMA-334) e ISO (ISO/IEC 23270). C# es uno de los lenguajes de programación diseñados para la infraestructura de lenguaje común. Su sintaxis básica deriva de C/C++ y utiliza el modelo de objetos de la plataforma .NET, similar al de Java, aunque incluye mejoras derivadas de otros lenguajes.



Figura 9: Logo del lenguaje de programación C#

Este software en la actualidad tiene una documentación abundante para poder comprender y utilizar al máximo sus prestaciones. Es por esto que el desarrollo de la aplicación de RA se basó en la utilización de la documentación presente en la web, además de los foros oficiales (figura 10).

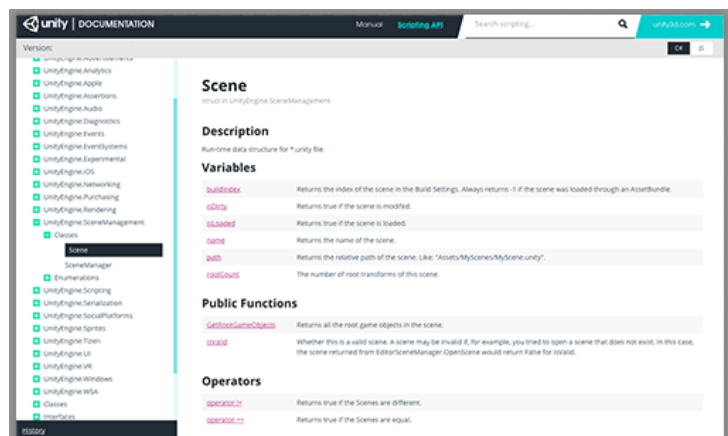


Figura 10: Unity Documentation

Creación de la aplicación

En una primera instancia se explica la interfaz que posee unity, explicando cada una de sus partes. La ventana del editor principal está compuesta por ventanas con pestañas que se pueden reorganizar, agrupar, separar y acoplar. Esto significa que el aspecto del editor puede ser diferente de un proyecto al otro, y de un desarrollador al siguiente, dependiendo en la preferencia personal y qué tipo de trabajo está haciendo.

El manual de la documentación oficial explica la disposición por defecto de Windows, mostrando las partes más generales en la ventana del editor:

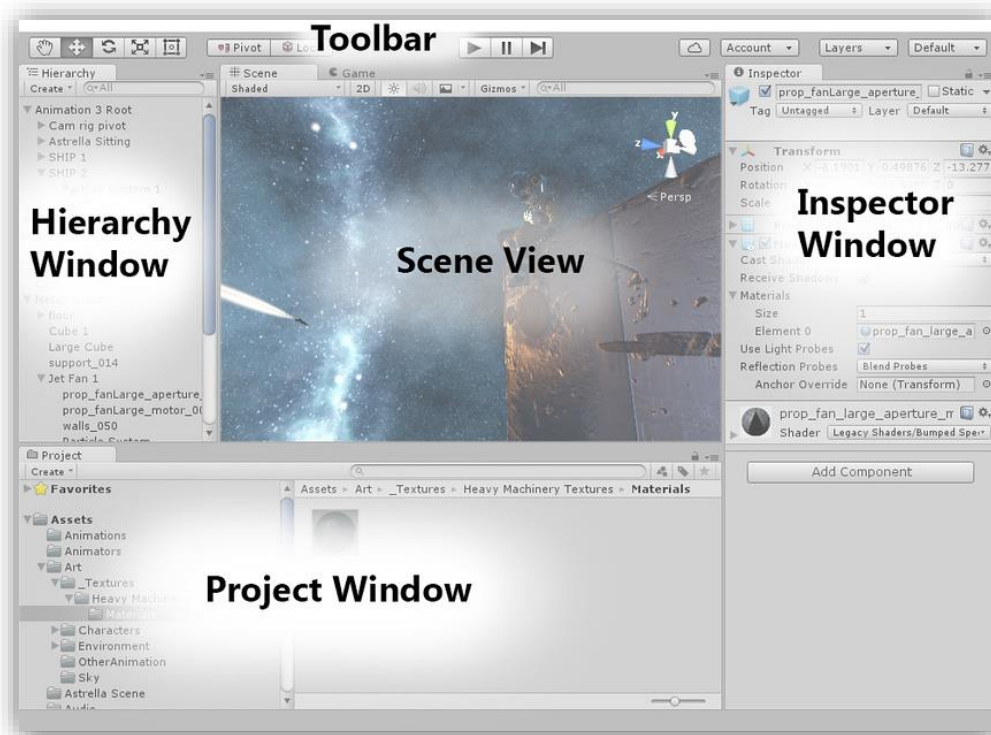


Figura 11: Partes de la ventana del editor de Unity.

- **The Project Window** (ventana del proyecto) muestra sus assets de librería que están disponibles para ser usados. Cuando usted importe sus assets a su proyecto, estos aparecen aquí. Un asset es una representación de cualquier item que puede ser utilizado en un juego o proyecto. Un asset podría venir de un archivo creado afuera de Unity, tal como un modelo 3D, un archivo de audio, una imagen, o cualquiera de los otros tipos de archivos que Unity soporta. También hay otros tipos de asset que pueden ser creados dentro de Unity, tal como un Animator Controller, un Audio Mixer o una Render Texture.
- **The Scene View** (vista de escena) le permite a usted una navegación visual y editar su escena. La scene view puede mostrar una perspectiva 2D o 3D dependiendo en el tipo de proyecto en el que esté trabajando. Aquí se ubican todos los objetos <GameObjects> que conforman la escena y donde se pueden manipular y editar.
- **The Hierarchy Window** (ventana de jerarquía) es una representación de texto jerárquico de cada objeto en la escena. Cada elemento en la escena tiene una entrada en la jerarquía, por lo que las dos ventanas están inherentemente vinculadas. La jerarquía revela la estructura de cómo los objetos están agrupados el uno al otro.

- **The Inspector Window** (ventana del inspector) le permite a usted visualizar y editar todas las propiedades del objeto actualmente seleccionado. Ya que diferentes objetos tienen diferentes propiedades, el layout (diseño) y contenido de la ventana del inspector va a variar.
- **The Toolbar** (barra de herramientas) proporciona un acceso a las características más esenciales para trabajar. En la izquierda contiene las herramientas básicas para manipular la scene view y los objetos dentro de esta. En el centro están los controles de reproducción, pausa, y pasos. Los botones a la derecha le dan acceso a sus servicios de Unity Cloud y su cuenta de Unity, seguido por un menú de visibilidad de capas, y finalmente el menú del layout del editor (que proporciona algunos diseños alternativos para la ventana del editor, y le permite a usted guardar sus propios layouts personalizados). La barra de herramienta no es una ventana, y solamente es parte de la interfaz de Unity que no se puede re-ajustar.

Creación de la interfaz de usuario

Unity ofrece una herramienta para crear una interfaz de usuario (UI) rápida e intuitiva, conocida como “Canvas”. El **Canvas** es el área donde todos los elementos UI deben estar. El Canvas es un Game Object con un componente Canvas en él, y todos los elementos UI deben ser hijos de dicho Canvas.

Creando un nuevo elemento UI, tal como una Image (imagen) utilizando el menú **GameObject > UI > Image**, automáticamente crea un Canvas si ya no hay uno en la escena ya. El elemento UI es creado como un hijo de este Canvas.

El área Canvas es mostrado como un rectángulo en la Vista de Escena. Esto lo hace fácil posicionar los elementos UI si necesitar tener una Vista de Juego todo el tiempo. Para la aplicación presentada se dispuso de la siguiente interfaz que se mostrará como pantalla principal en cada dispositivo:



Figura 12: Interfaz de usuario .

La interfaz mostrada previamente en la figura 12 está compuesta por elementos como botones “ANIMACION” e “INFO” destinados a capturar eventos y así poder desplegar una animación y mostrar información respectivamente. Además se agrega un botón destinado a encender el flash en el caso de necesitarlo. Para una interacción más fluida se dispuso de un cuadro de texto entre corchetes para orientar al usuario en el momento de utilizar la app. Los dos objetos que se agregan en la parte inferior son objetos pertenecientes a un tipo particular de GameObject, en este caso un Joystick en donde se carga del lado derecho una “palanca virtual” que permitirá poder manipular los objetos que se desplieguen en la pantalla, y del lado izquierdo un “slider” con la función de poder realizar un acercamiento/alejamiento del objeto.

El objeto “Joystick virtual” se descargó de la tienda oficial de asset que ofrece Unity (“Asset Store”) totalmente gratis, aunque por supuesto se pueden encontrar diferentes objetos mucho más avanzados con algún costo en dólares. Una vez descargado, se agrega al proyecto en la carpeta de asset como cualquier otro objeto.

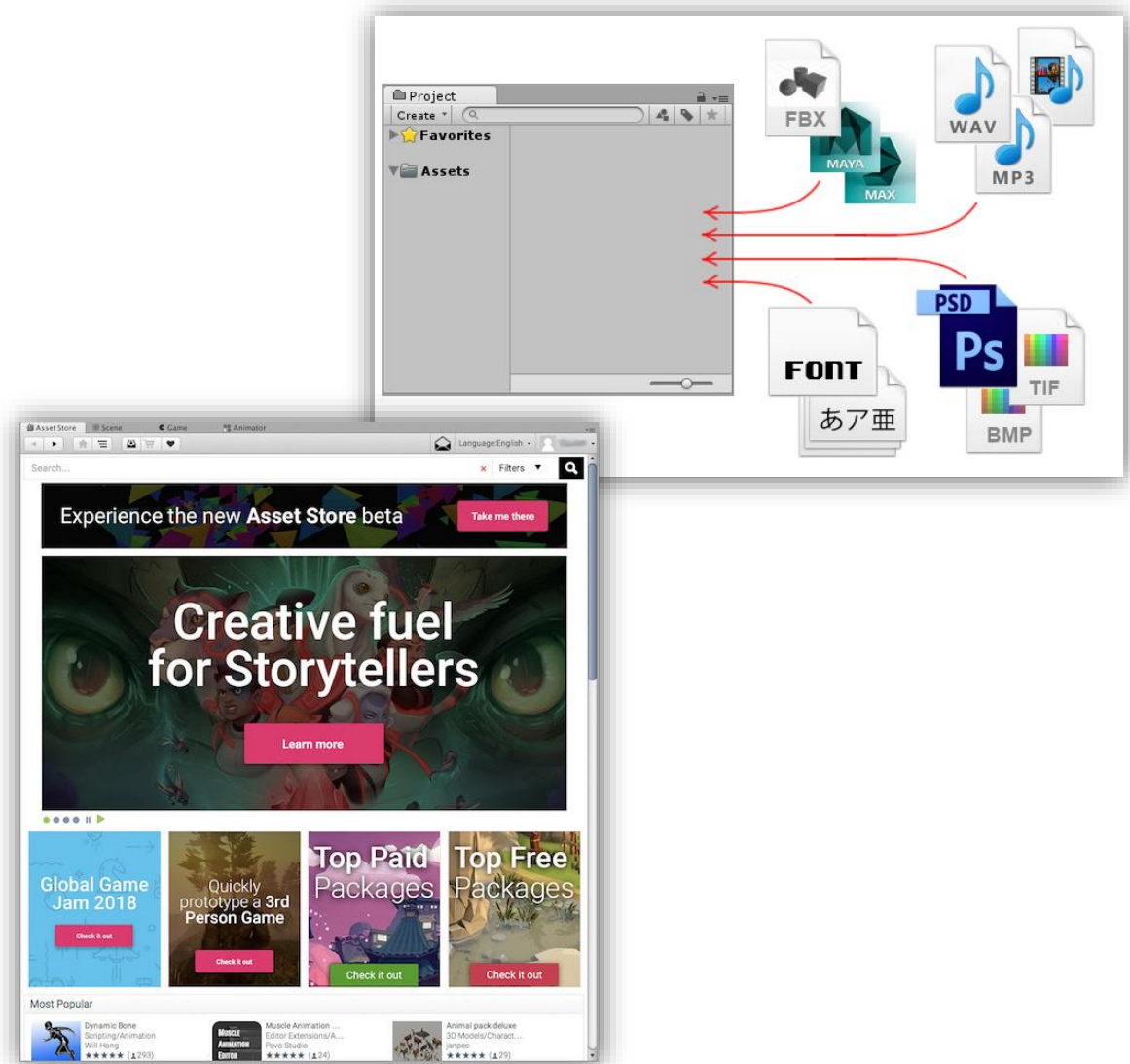


Figura 13: Assets en un proyecto y Asset Store en Unity

Todas las funcionalidades de estos componentes explicados se profundizarán más adelante en el presente informe en la sección de scripts de programación.

Objetos en 3D y su modificación en Blender

Como se explicó en la sección anterior en Unity es posible agregar diferentes objetos para ser mostrados en una escena como un gameobject. En este caso se utilizaron objetos en 3D de diferentes partes del cuerpo, descargados de una página que ofrece modelos con extensión .obj o .fex o .max para luego ser impresos o utilizados en cualquier proyecto de manera gratuita (figura 14).

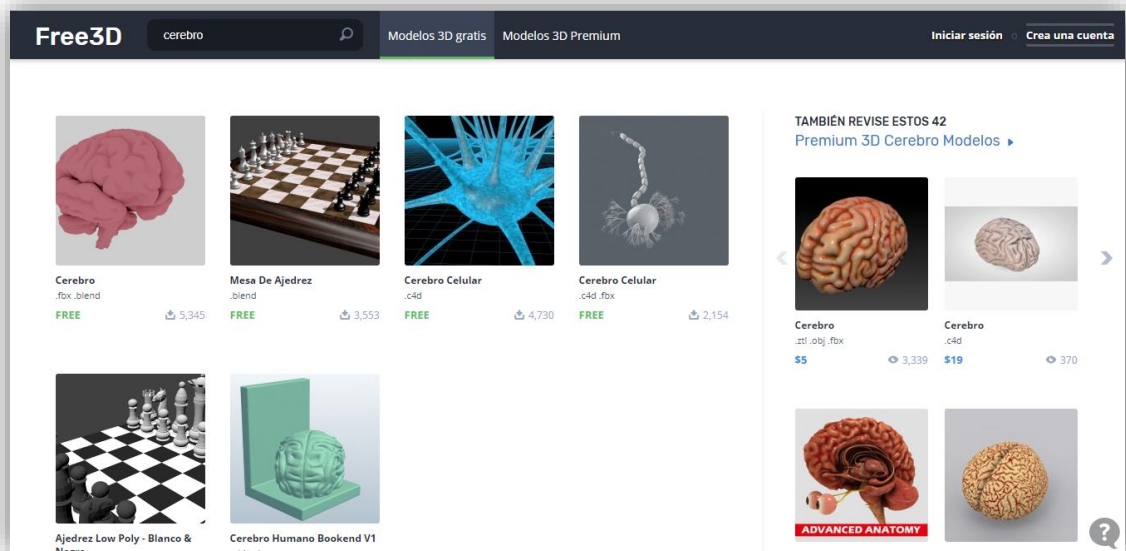


Figura 14: Free3D pagina para descargar objetos 3D gratuitos.

Una vez descargados los archivos requeridos se pasa a utilizar un software que permite modificar diferentes extensiones de archivos que contienen objetos en tres dimensiones. El software mencionado es blender que es un programa informático multi plataforma, dedicado especialmente al modelado, iluminación, renderizado, animación y creación de gráficos tridimensionales. También de composición digital utilizando la técnica procesal de nodos, edición de vídeo, escultura y pintura digital. Además es importante aclarar que es de código abierto, es decir, totalmente gratis y que cuente con una extensa documentación oficial.

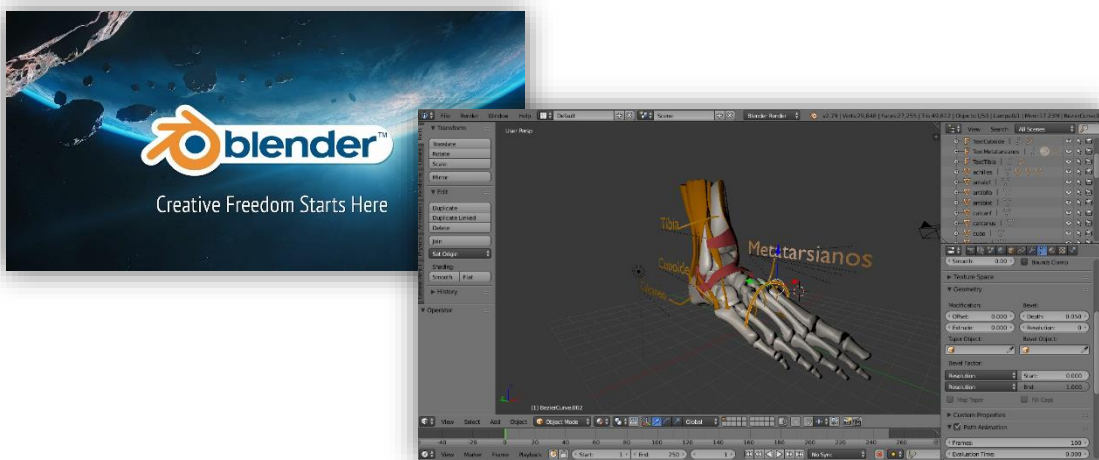


Figura 15: Blender, software para manipulación de objetos tridimensionales.

Esta herramienta informática fue de gran ayuda para poder incorporar información a cada objeto en 3D para luego crear un solo archivo con extensión .blend, totalmente compatible en Unity. En particular se optó por el despliegue de 4 objetos a los cuales se les agrega información a cada una de las partes que componen a los objetos.

- Huesos del pie
- Corazón
- Cráneo
- Vértebra Lumbar

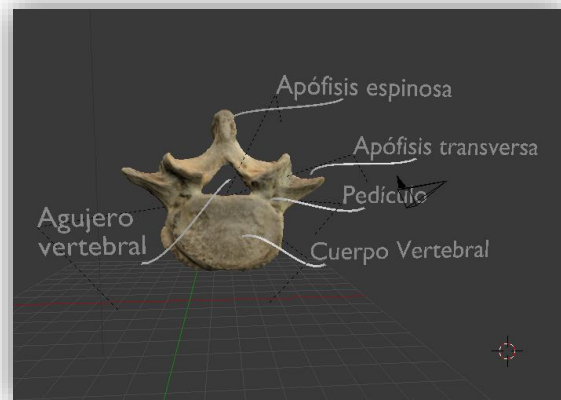
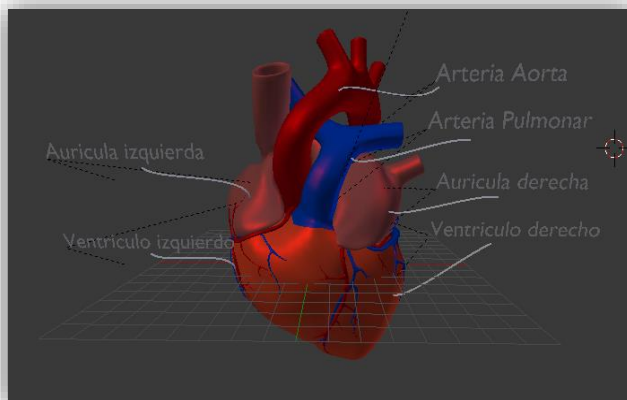
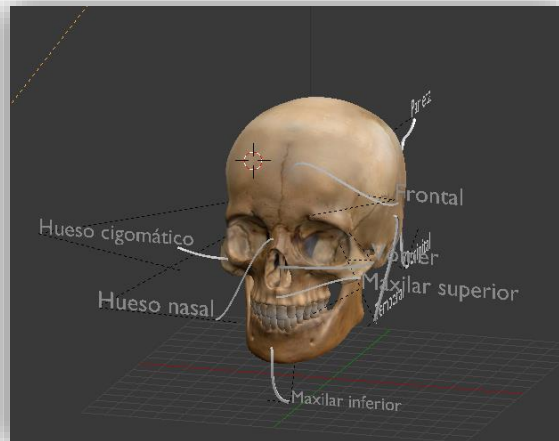


Figura 16: Modelos en 3D de partes del cuerpo humano modificados en blender para ser utilizados en unity. .

Además de la información que fue agregada en cada parte, se adicionó el tipo de material para la superficie del objeto, es decir un tipo de render. Con estos objetos listos para ser mostrados, importan como asset para el proyecto general.

Vuforia y Android

Vuforia es una plataforma de desarrollo de aplicaciones de Realidad Aumentada (AR) y Realidad Mixta (MR) multiplataforma, con seguimiento robusto y rendimiento en una variedad de hardware (incluyendo dispositivos móviles y monitores de realidad mixta montados en la cabeza (HMD). Utiliza la tecnología Computer Vision para reconocer y rastrear imágenes planas (objetivos de imagen) y objetos 3D simples, como cuadros, en tiempo real.

Esta capacidad de registro de imágenes permite a los desarrolladores ubicar y orientar objetos virtuales, como modelos 3D y otros medios, en relación con imágenes del mundo real cuando se visualizan a través de la cámara de un dispositivo móvil. El objeto virtual luego rastrea la posición y orientación de la imagen en tiempo real para que la perspectiva del espectador en el objeto se corresponde con su perspectiva en el objetivo de imagen, por lo que parece que el objeto virtual es una parte de la escena del mundo real.

La integración de Unity en Vuforia le permite crear aplicaciones y juegos de visión para Android e iOS utilizando un flujo de trabajo de creación de arrastrar y soltar.



Figura 17: Logo de Vuforia

Además para el desarrollo de una aplicación de Android es necesario un SDK (Software Development Kit) de Android, el cual incluye un conjunto de herramientas de desarrollo. Comprende un depurador de código, biblioteca, documentación, ejemplos de código y tutoriales. Las plataformas de desarrollo soportadas incluyen GNU/Linux, Mac OS X 10.5.8 o posterior, y Windows XP o posterior. La plataforma integral de desarrollo (IDE, Integrated Development Environment) soportada oficialmente es Android Studio que permite la descarga y la instalación del SDK de Android en el ordenador. Con estos componentes (vuforia SDK y android SDK) Unity es capaz de utilizarlos para crear aplicaciones de realidad virtual que corran en diferentes dispositivos móviles.



Figura 18: Integración de Android y Vuforia en Unity para crear una app.

Marcadores y creación de base de datos en Vuforia

Para poder utilizar el SDK de Vuforia, es necesario crear una cuenta en la página <https://developer.vuforia.com> ya que Unity nos pide una clave de activación. Una vez creada la cuenta, se ingresa y se elige la pestaña **Develop**, luego la pestaña **Target Manager** y se presiona el botón **Add Database**. Al presionar el botón se despliega un cuadro en donde se asigna el nombre de la Base de datos y se debe seleccionar el casillero **Device**. Una vez creada la base de datos, se ingresa y se presiona el botón **Add Target** para crear un nuevo marcador. Se despliega un cuadro en donde se puede seleccionar qué tipo de marcador se desea crear (si se desea que la cámara reconozca una imagen simple, una imagen que cubre un cubo, un cilindro, o un cuerpo 3D). Se selecciona la primera opción, luego se busca el archivo de la imagen que tendrá el marcador, y se coloca el tamaño y el nombre que tendrá dentro de Unity.

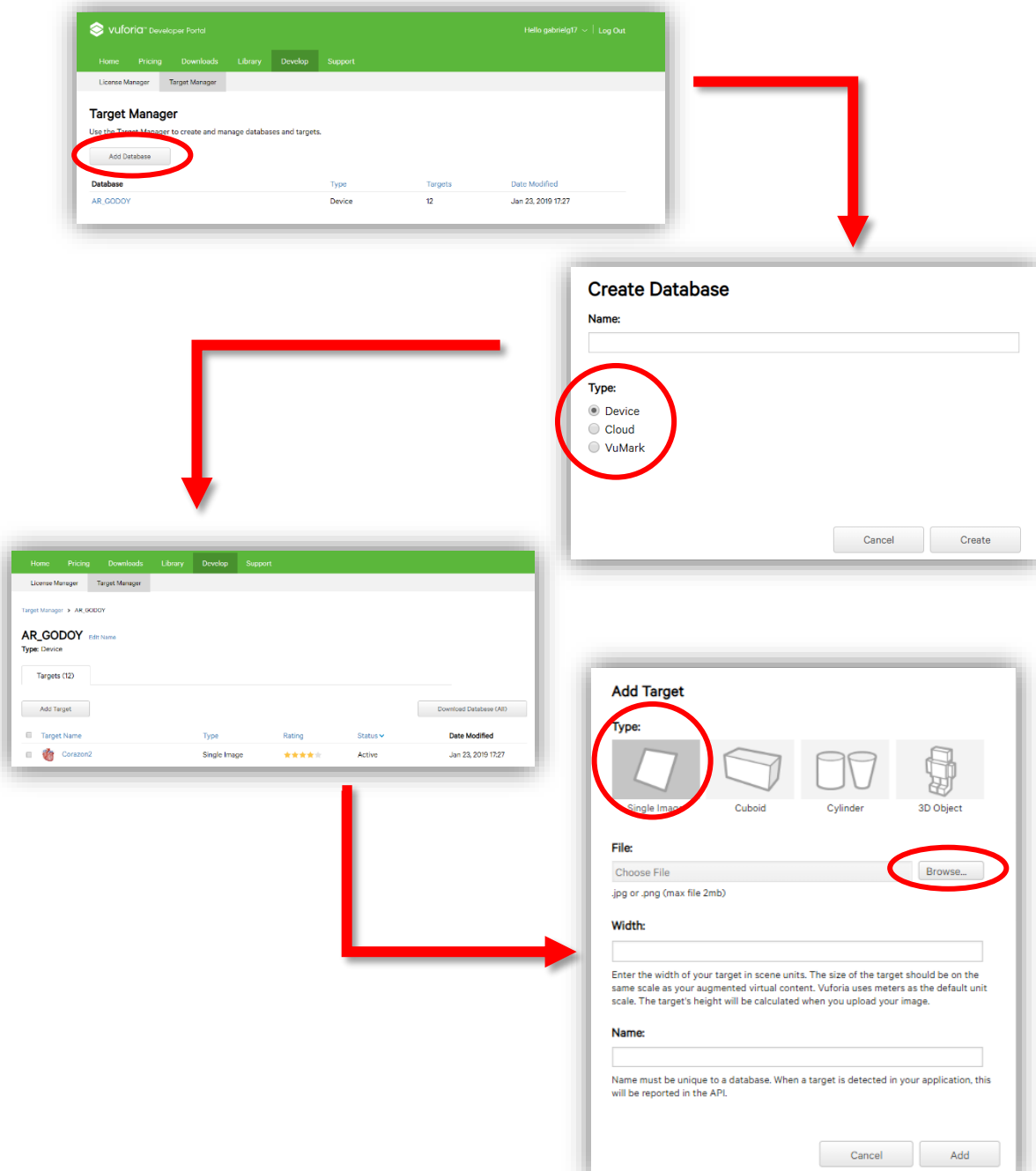


Figura 19: Pasos para crear una base de datos y agregar marcadores.

Incorporación de marcadores a Unity

Una vez creados los cinco marcadores, se descarga la base de datos presionando el botón Download Database y se importa desde Unity ingresando en la barra de herramientas y seleccionado **Assets -> Import Package -> Custom Package**.

Se debe seguir el siguiente procedimiento:

- Se descarga e instala el complemento de Vuforia para Unity.
- Se habilita el complemento ingresando a **File -> Build Settings -> Player Setting** y luego en el inspector ingresando a **XR Settings**, se tilda el casillero **Vuforia Augmented Reality**.
- Se incorpora una **<AR Camera>** a la escena ingresando en la pestaña de barra de herramientas **GameObject -> Vuforia -> AR Camera**. Se debe eliminar la cámara que se encontraba anteriormente para evitar inconvenientes.
- En la página de Vuforia, se selecciona la pestaña **Develop**, luego la pestaña **License Manager** y se copia el código que aparece.
- Nuevamente en Unity, se selecciona la AR Camera y en el inspector se presiona **Open Vuforia configuration**.
- En la ranura **App License Key** se pega el código de la licencia.
- Se tilda el casillero **Load Object Targets**.

Una vez realizado lo anterior ya se puede colocar los marcadores en la escena. Se incorpora cada marcador a la escena ingresando en la pestaña de barra de herramientas **GameObject -> Vuforia -> Image**. Los marcadores seleccionados fueron los siguientes:

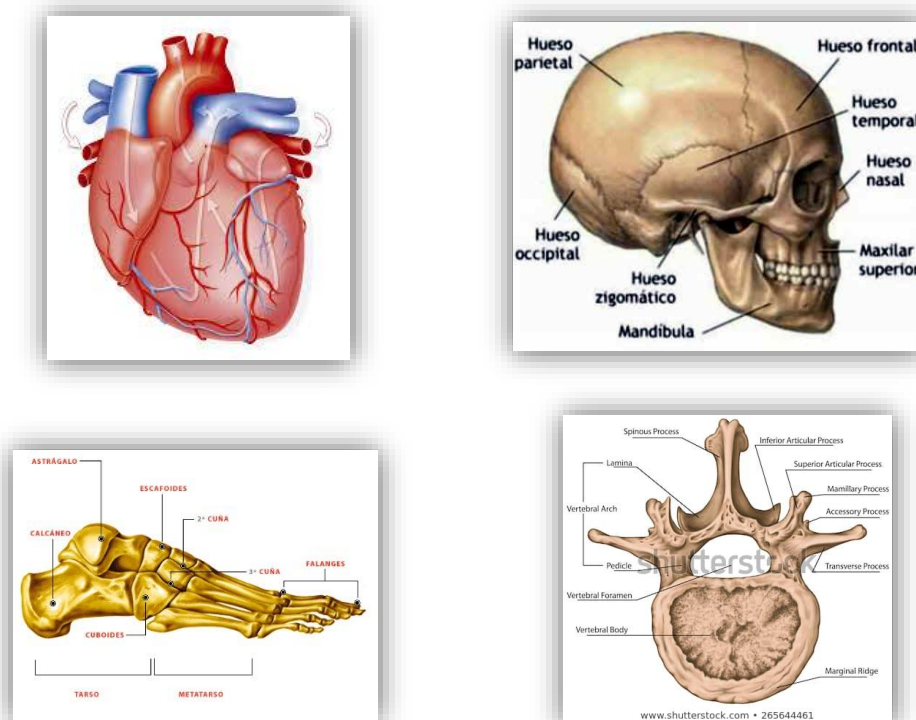


Figura 20: Target que funcionan como marcadores para realidad aumentada en una base de datos de vuforia.

Con estos marcadores agregados en la escena de Unity, se procedió a asociar a cada objeto a mostrar con su respectivo **Target Image**, esto se logra seleccionando el objeto de la ventana de proyecto y arrastrándolo hasta la ventana de jerarquía. Esta acción convierte a los objetos en 3D en GameObject del proyecto, con sus respectivos componentes.

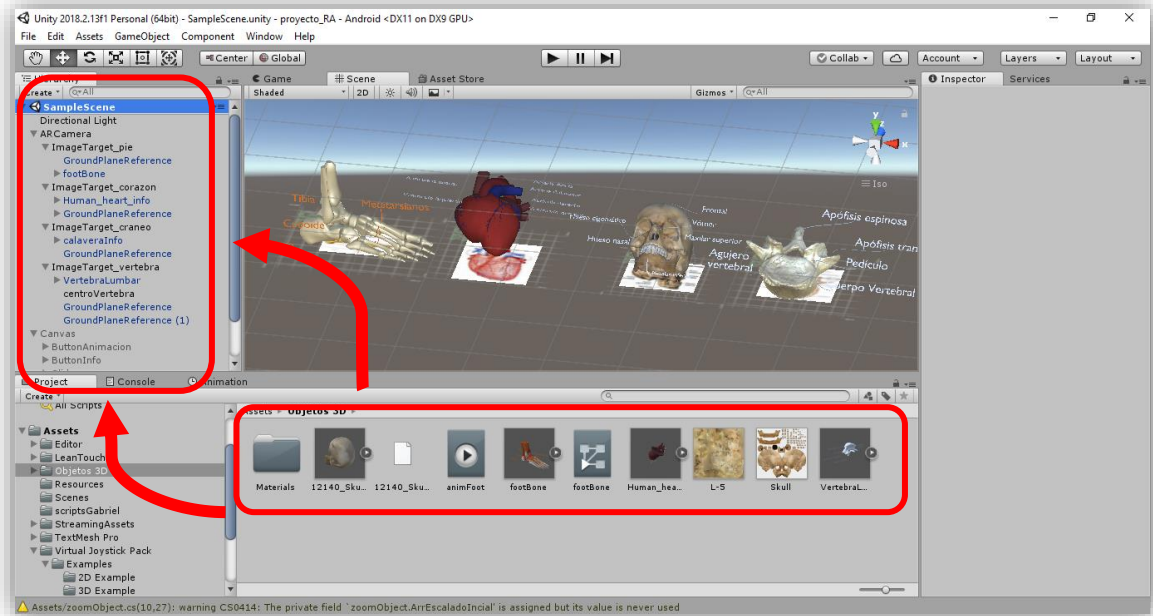


Figura 21: Asociación de objetos de la ventana de proyecto a la ventana de jerarquía de la escena.

Scripts

Teniendo todos los GameObject necesarios para el proyecto, en este caso las partes del cuerpo humano (con su respectiva información) asociados a cada marcador, se procede a generar el código de programación que permitirá dotar de funciones a los distintos elementos del **Canvas**. Esto se logra agregando un nuevo script en cada componente en la **ventana inspector-> addComponent-> New Script**.

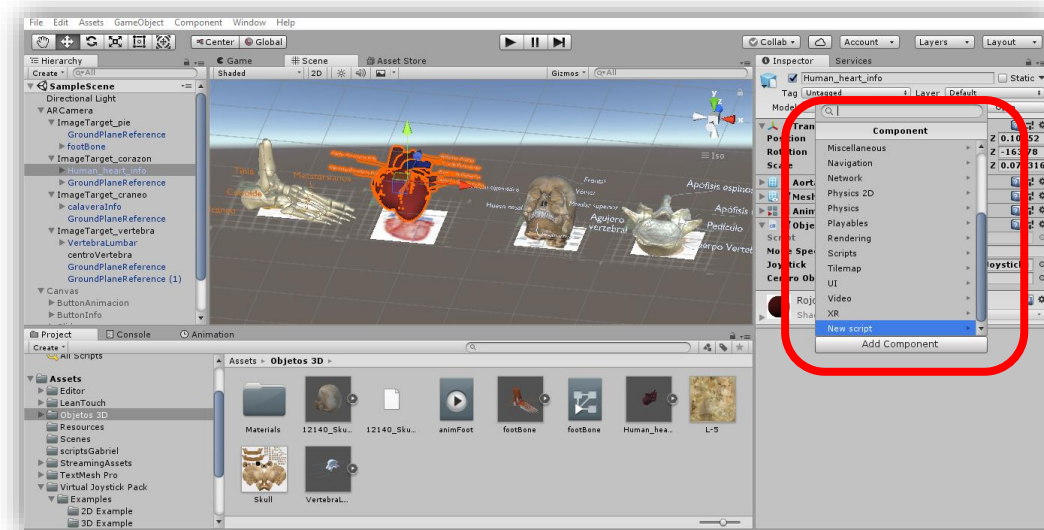


Figura 22: Creación de un nuevo Script

Para el objetivo de la aplicación fue necesario la generación de 5 scripts:

- buttonActionInfo
- activeLedCamera
- objectMove
- zoomObject
- animationController

buttonActionInfo :

Para este script se programa con 4 atributos públicos, en este caso un arreglo de GameObject que representan las partes que contienen la información de cada parte de los modelos en 3D. En la función void Start() (que es la primer función que se ejecuta al correr el script) se desactivan todos los gameobject es decir la información de los objetos. Luego se establece una función activarInfo() que se activa en la ventana de inspección del botón "INFO".

```

using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class buttonActionInfo : MonoBehaviour {

    public GameObject[] Info_Craneio = new GameObject[9];
    public GameObject[] Info_Corazon = new GameObject[6];
    public GameObject[] Info_Pie = new GameObject[3];
    public GameObject[] Info_Vertebra = new GameObject[5];

    private Button boton;
    private bool estadoInfo;
}

```

```

void Start(){
    //desactivo la info de cada objeto (arreglo de objetos)
    for (int i = 0; i < Info_Craneio.Length; i++){
        Info_Craneio[i].SetActive(false);
    }

    for (int i = 0; i < Info_Corazon.Length; i++){
        Info_Corazon[i].SetActive(false);
    }

    for (int i = 0; i < Info_Pie.Length; i++){
        Info_Pie[i].SetActive(false);
    }

    for (int i = 0; i < Info_Vertebra.Length; i++){
        Info_Vertebra[i].SetActive(false);
    }

    estadoInfo = false;
}

```

```

public void activarInfo() {
    if (estadoInfo == false)
    {
        for (int i = 0; i < Info_Craneio.Length; i++){
            Info_Craneio[i].SetActive(true);
        }
        for (int i = 0; i < Info_Corazon.Length; i++){
            Info_Corazon[i].SetActive(true);
        }
        for (int i = 0; i < Info_Pie.Length; i++){
            Info_Pie[i].SetActive(true);
        }
        for (int i = 0; i < Info_Vertebra.Length; i++){
            Info_Vertebra[i].SetActive(true);
        }
        estadoInfo = true;
    }
    else {
        for (int i = 0; i < Info_Craneio.Length; i++){
            Info_Craneio[i].SetActive(false);
        }
        for (int i = 0; i < Info_Corazon.Length; i++){
            Info_Corazon[i].SetActive(false);
        }
        for (int i = 0; i < Info_Pie.Length; i++){
            Info_Pie[i].SetActive(false);
        }
        for (int i = 0; i < Info_Vertebra.Length; i++){
            Info_Vertebra[i].SetActive(false);
        }
        estadoInfo = false;
    }
}

```

Figura 23: Script "buttonActionInfo.cs"

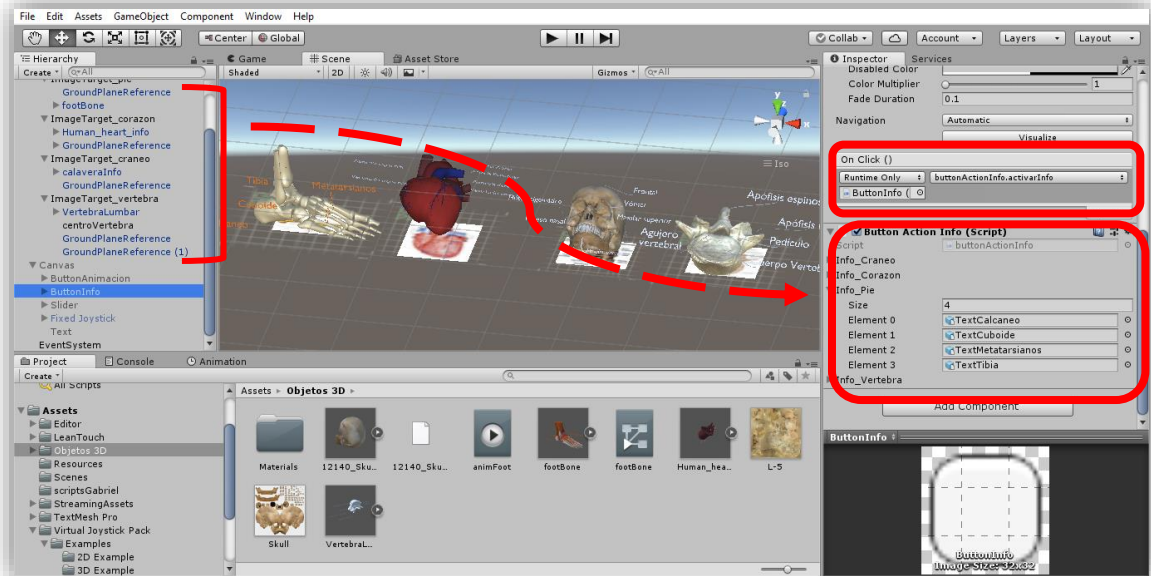


Figura 24: Activación del script en los componentes del boton INFO, arrastrando los game object presentes en la escena

objectMove:

Aquí es donde se programa el algoritmo para la manipulación, en este caso la rotación del objeto en 3D. En este script se programa la función de la palanca del joystick incorporado un algoritmo tanto en la función Start() para obtener por primera vez el centro del objeto que se va a rotar y luego en la función Update() se gira el objeto al detectar un cambio en la palanca y modificando el componente transform.position del GameObject. Luego de la misma manera que el caso anterior se incorpora en la ventana inspector de cada gameobject que se quiera manipular.

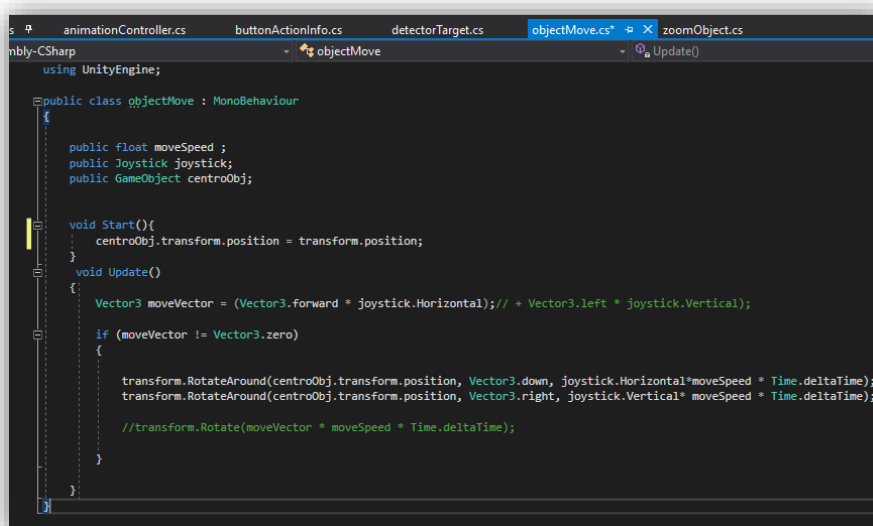
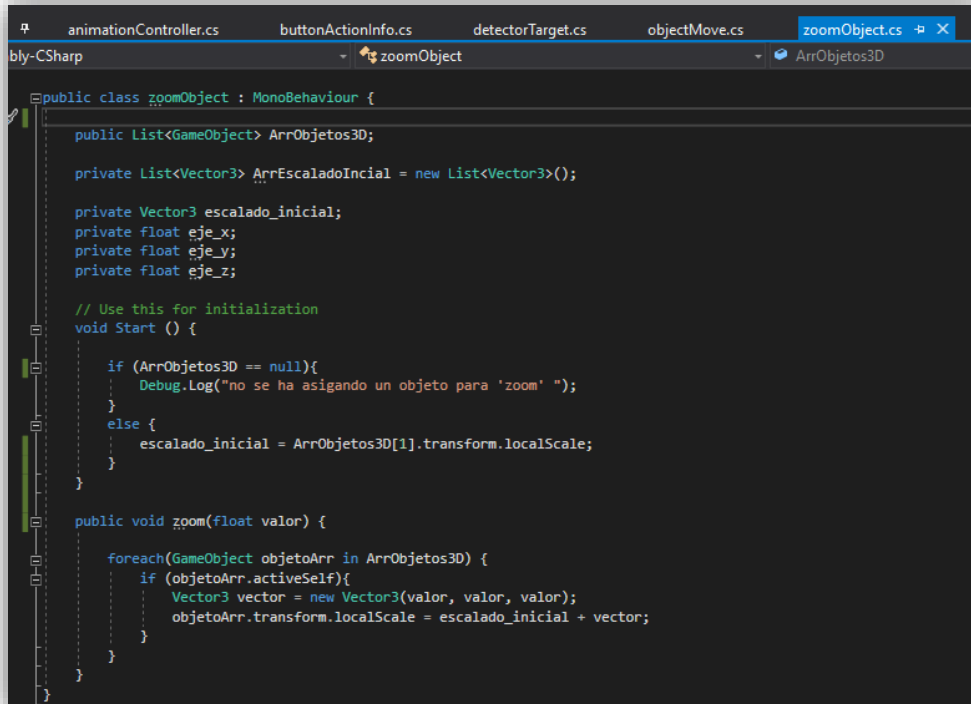


Figura 25: Script objectMove para incorporar a cada GameObject a manipular.

zoomObject :

Este script es para poder acercar y alejar los objetos de la pantalla, en otras palabras generar un zoom en cada GameObject. Para esto fue necesario programar que el script reciba una lista de objetos a manipular. La función más importante codificada es la función zoom() que modifica el componente transform.localScale de cada GameObject. Este Script se agrega como componente del elemento slider del canvas.



```

public class zoomObject : MonoBehaviour {
    public List<GameObject> ArrObjetos3D;

    private List<Vector3> ArrEscaladoInicial = new List<Vector3>();

    private Vector3 escalado_inicial;
    private float eje_x;
    private float eje_y;
    private float eje_z;

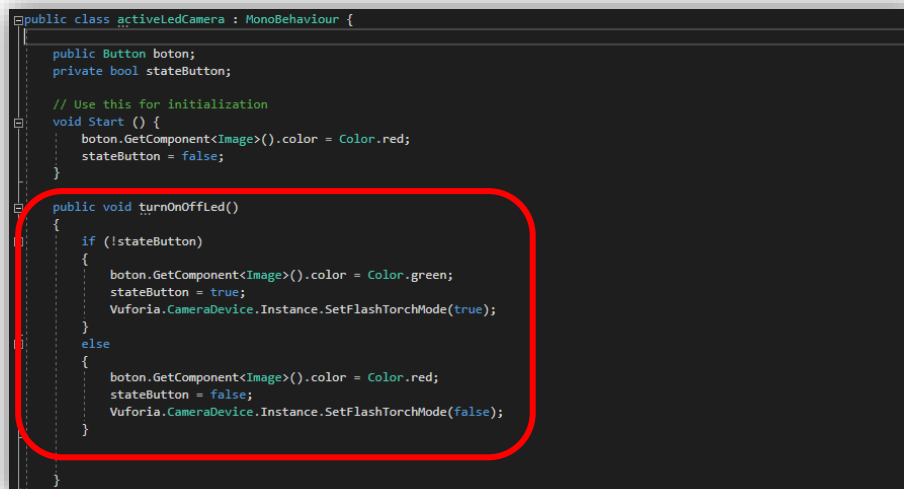
    // Use this for initialization
    void Start () {
        if (ArrObjetos3D == null){
            Debug.Log("no se ha asignado un objeto para 'zoom' ");
        }
        else {
            escalado_inicial = ArrObjetos3D[1].transform.localScale;
        }
    }

    public void zoom(float valor) {
        foreach(GameObject objetoArr in ArrObjetos3D) {
            if (objetoArr.activeSelf){
                Vector3 vector = new Vector3(valor, valor, valor);
                objetoArr.transform.localScale = escalado_inicial + vector;
            }
        }
    }
}
  
```

Figura 26: Script zoomObject para incorporar a cada GameObject a manipular.

ActiveLedCamera:

Este script permite encender el led utilizado como flash del dispositivo, para obtener una mejor iluminación y así poder detectar el marcador. Además cuando es presionado el botón cambia de color Rojo a verde. Todo gracias a la función turnOnOffLedCamera.



```

public class activeLedCamera : MonoBehaviour {
    public Button boton;
    private bool stateButton;

    // Use this for initialization
    void Start () {
        boton.GetComponent<Image>().color = Color.red;
        stateButton = false;
    }

    public void turnOnOffLed()
    {
        if (!stateButton)
        {
            boton.GetComponent<Image>().color = Color.green;
            stateButton = true;
            Vuforia.CameraDevice.Instance.SetFlashTorchMode(true);
        }
        else
        {
            boton.GetComponent<Image>().color = Color.red;
            stateButton = false;
            Vuforia.CameraDevice.Instance.SetFlashTorchMode(false);
        }
    }
}
  
```

Figura 27: Script activeLedCamera para encender flash del dispositivo.

animationController:

Previo a realizar el código que permita activar una animación, es necesario crear previamente una animación. Esto se logra creando en la pestaña Animation a la derecha de la ventana proyecto. Aquí se selecciona el gameObject a animar y se modifica el componente necesario, por ejemplo en el caso de la animación del corazón, este solo se modifica la escala del objeto, provocando una especie de latido del mismo.

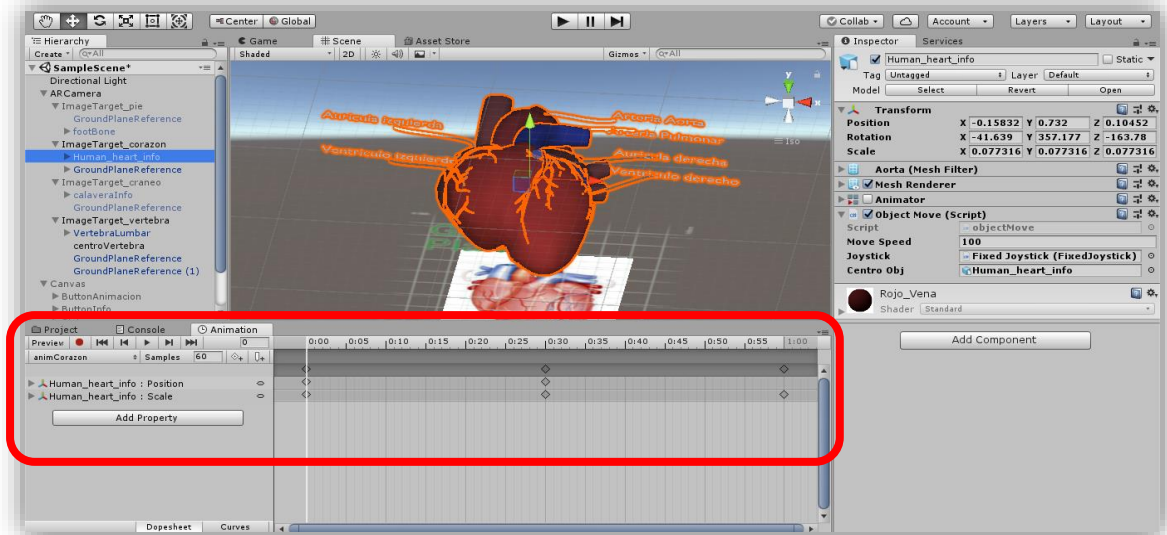


Figura 28: Creacion de animacion en Unity.

De la misma manera se genera una animación para cada objeto presente en el proyecto. Luego de esto se procesa a crear el script para poder activar la animación de cada elemento. Esta porción de código está destinada a recibir un arreglo de GameObjects para tomar su componente "animator" y activarlo o desactivarlo cada vez que se presione el botón animación. Este script es agregado en los componentes del botón Animación.

```
public class animationController : MonoBehaviour {
    public GameObject[] objToAnimation;
    private Animator[] animacion;

    void Start()
    {
        for (int i = 0; i < objToAnimation.Length; i++)
        {
            animacion[i] = objToAnimation[i].GetComponent<Animator>();
            animacion[i].enabled = false;
        }
    }

    public void enableAnimation()
    {
        for (int i = 0; i < objToAnimation.Length; i++)
        {
            if (objToAnimation[i].activeSelf)
            {
                animacion[i].enabled = true;
            }
            else {
                animacion[i].enabled = false;
            }
        }
    }
}
```

Figura 29: Script animationController.

Exportación de la aplicación.

Luego de realizar diferentes pruebas corriendo en el ordenador donde se crea la aplicación (utilizando la cámara y el mouse). Se procesa a compilar el código generado para poder exportar la aplicación a un dispositivo Android.

Para esto se requiere tener previamente el SDK de android instalado en el ordenador, para compilar. Entonces el camino a seguir es **File-> Build Setting -> Seleccionar plataforma Android** y en la ventana inspector seleccionar la opción de **AR augmented reality Vuforia**.

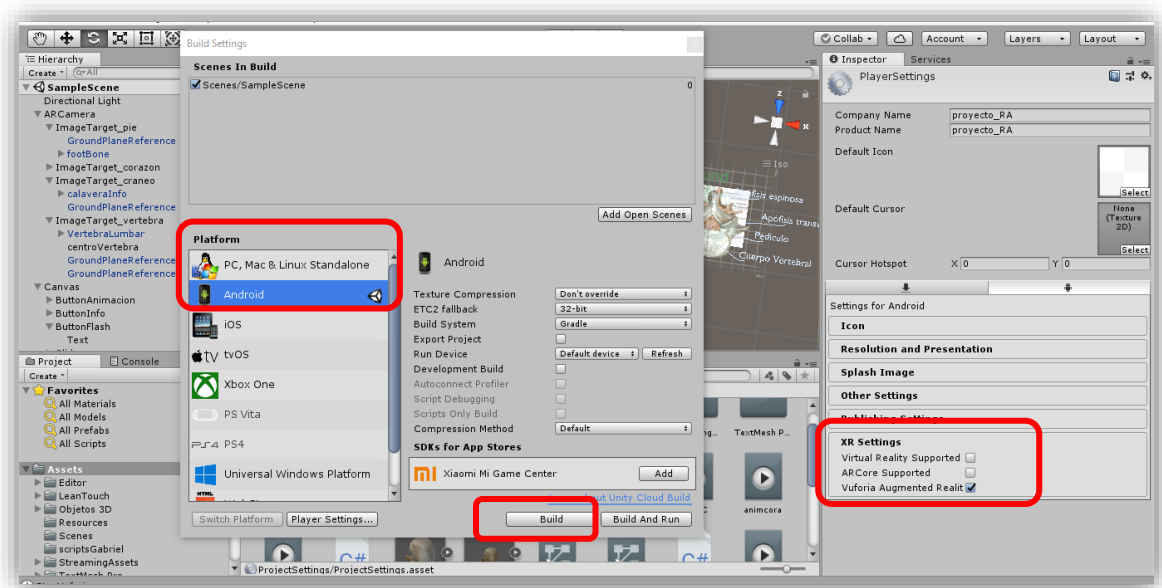


Figura 30: Compilación de aplicación en android con Unity 3D..

Además es posible indicarle la versión mínima de Android soportada, asignar un icono específico para la app entre otras especificaciones. Al finalizar la operación Unity crea un archivo .apk que al ser copiado a la memoria de un dispositivo Android permite instalar la aplicación de manera sencilla y rápida.

Resultados

El dispositivo para realizar las pruebas de la aplicación fue el Smartphone Motorola moto C, el cual es un equipo de gama media-baja con características básicas comunes y con una versión del sistema operativo Android 7.0 (Nugat). Además se armaron una serie de hojas que contienen información de cada uno de los objetos a representar y el marcador propiamente ubicado (ver anexo). Estas hojas simularían paginas de una enciclopedia que contenga toda la información necesaria y el marcador para poder activar la aplicación de realidad aumentada.

Una vez instalada la aplicación se da paso a probar las funcionalidades de cada elemento de la interfaz, como son los botones, el joystick y el slider. Para dejar registro se realizaron capturas de pantalla desde el dispositivo con la aplicación en ejecución.



Figura 31: Dispositivo android para pruebas (moto c).

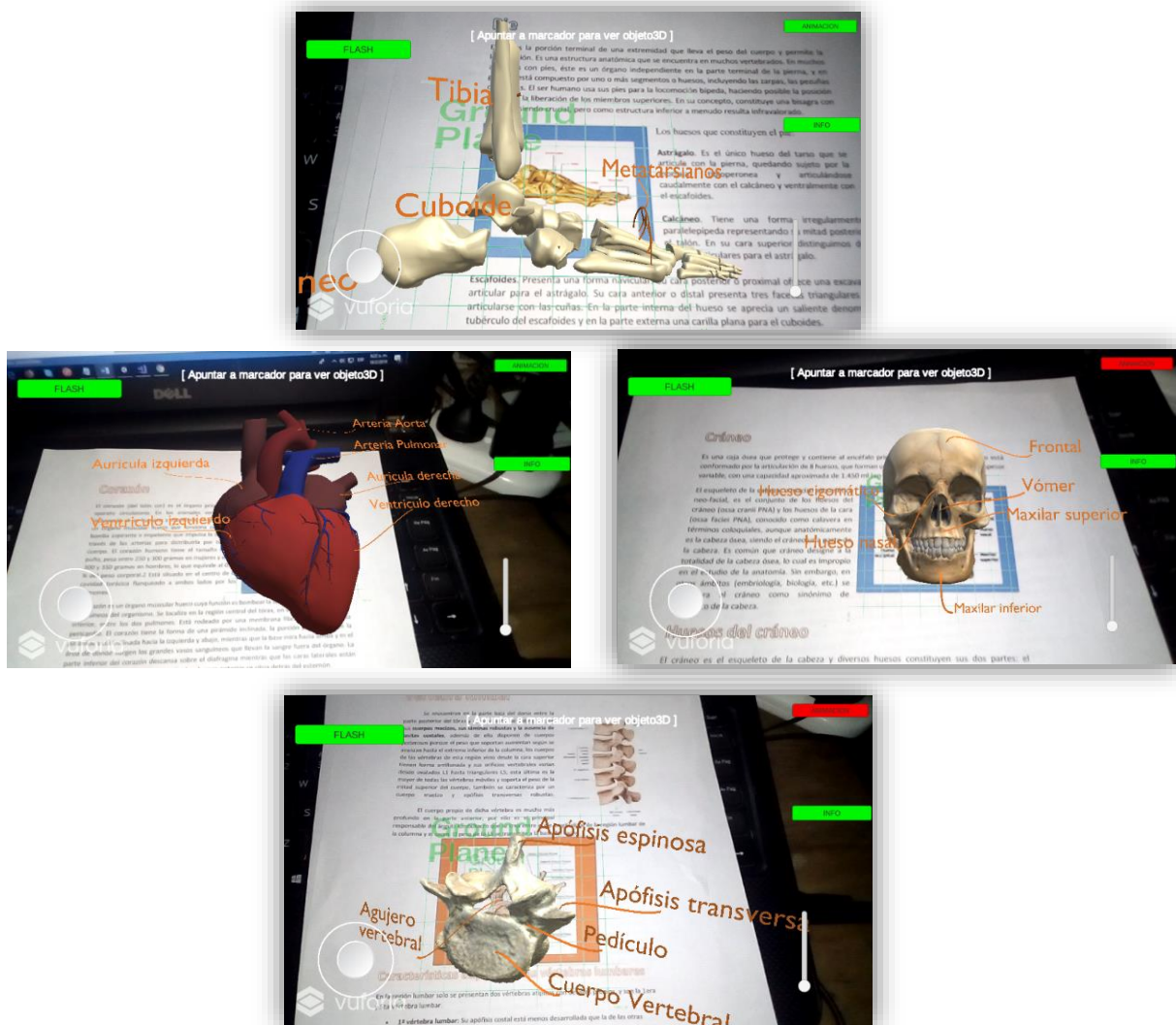


Figura 32: Capturas de pantalla de la aplicación ejecutándose en un dispositivo android

Conclusiones y Mejoras a futuro

Con las pruebas realizadas se logra llegar al objetivo de poder desplegar un objeto en 3D, en este caso una parte u órgano del ser humano, manipularlo y mostrar animaciones. Gracias a la grandes prestaciones que ofrece el motor de videojuegos utilizado (Unity) fue posible crear una aplicación para un dispositivo android de una manera sencilla y rápida y además sumamente confiable se logró instalar y ejecutar dicha app en un dispositivo de bajas prestaciones sin ningún inconveniente.

Esta aplicación queda como una potencial herramienta para poder ser utilizada al momento de aprender sobre el cuerpo humano, ya que puede ser introducida en el ámbito académico en donde en la actualidad la mayoría de los estudiantes tiene un Smartphone con sistema operativo Android funcionando. Además constituye una aplicación intuitiva y fácil de manejar para cualquier tipo de usuario ya que no requiere de conocimientos técnicos previos.

Las mejoras a futuro que se plantean son las de poder incorporar más objetos de la anatomía humana y mucho más representativos. Se plantea a futuro el uso de sonidos para una mejor interacción así como también la posibilidad de poder subdividir en partes el objeto desplegado en 3D para darle control total al usuario a la hora de manipularlo. Todo esto permitiría aún más sumergir al usuario en un mundo virtual mezclado con el real, logrando una experiencia enriquecedora en el momento del aprendizaje.

Referencias:

<https://free3d.com/es/modelos-3d> (Página para descargar modelos 3D gratuitos)

<https://docs.unity3d.com/es/current/Manual/UnityManual.html> (documentacion oficial Unity)

<https://www.vuforia.com/> (página oficial de Vuforia)

<https://docs.blender.org/> (página oficial de blender)

<https://developer.android.com/studio/command-line/adb> (para instalar android SDK)

<https://developer.vuforia.com/forum/unity-extension-technical-discussion/turn-onoff-led-flash-camera-unity-android> (para resolver encendido de led)

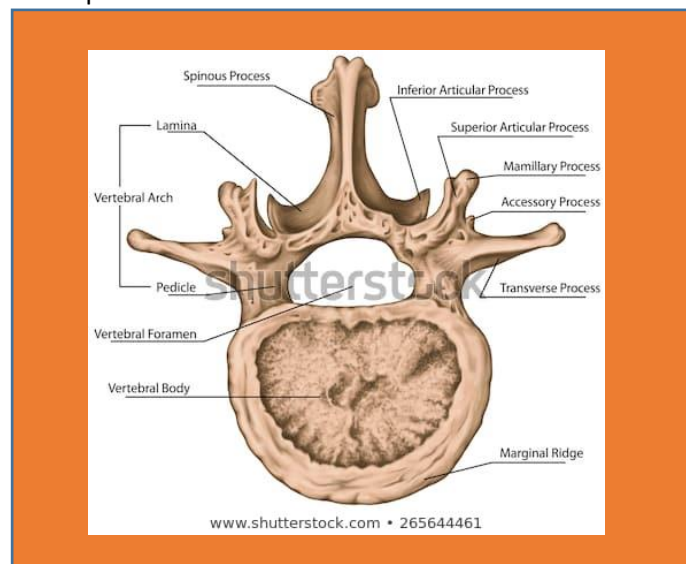
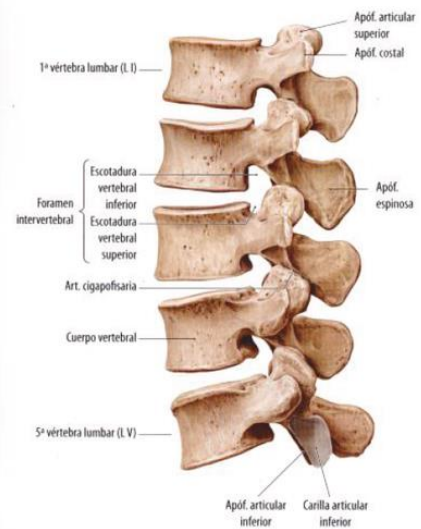
ANEXO

Hojas para realizar pruebas que representarían a una enciclopedia que brinda información textual así como la incorporación de marcadores para realidad aumentada.

Vértebra lumbar

Se encuentran en la parte baja del dorso entre la parte posterior del tórax y el sacro, estas se distinguen por sus **cuerpos macizos, sus láminas robustas y la ausencia de fositas costales**, además de ello disponen de cuerpos poderosos porque el peso que soportan aumentan según se avanzan hasta el extremo inferior de la columna, los cuerpos de las vértebras de esta región visto desde la cara superior tienen forma arriñonada y sus orificios vertebrales varían desde ovalados L1 hasta triangulares L5, esta última es la mayor de todas las vértebras móviles y soporta el peso de la mitad superior del cuerpo, también se caracteriza por un cuerpo macizo y apófisis transversas robustas.

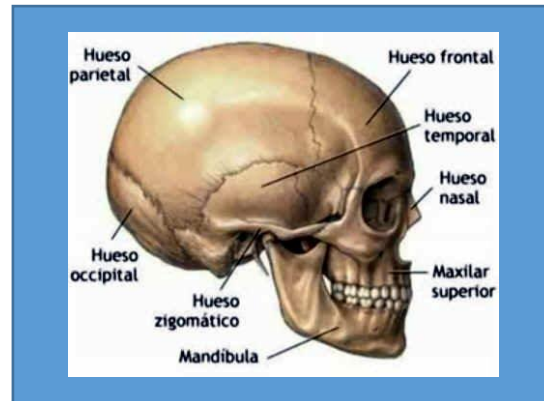
El cuerpo propio de dicha vértebra es mucho más profundo en la parte anterior, por ello es el principal responsable del ángulo lumbosacro que se crea entre el eje longitudinal de la región lumbar de la columna y el sacro. El peso de la L5 se transmite a la base del sacro.



Cráneo

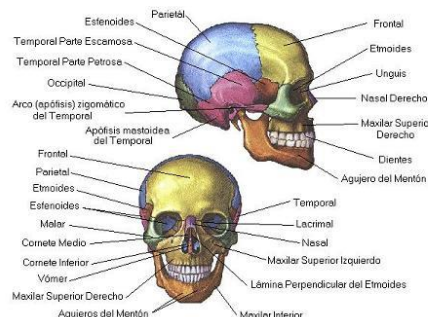
Es una caja ósea que protege y contiene al encéfalo principalmente. El cráneo humano está conformado por la articulación de 8 huesos, que forman una cavidad abierta y ovoide de espesor variable, con una capacidad aproximada de 1.450 ml (en adultos).

El esqueleto de la cabeza, o macizo esquelético neo-facial, es el conjunto de los huesos del cráneo (ossa cranii PNA) y los huesos de la cara (ossa faciei PNA), conocido como calavera en términos coloquiales, aunque anatómicamente es la cabeza ósea, siendo el cráneo una parte de la cabeza. Es común que cráneo designe a la totalidad de la cabeza ósea, lo cual es impropio en el estudio de la anatomía. Sin embargo, en otros ámbitos (embriología, biología, etc.) se considera el cráneo como sinónimo de esqueleto de la cabeza.



Huesos del cráneo

El cráneo es el esqueleto de la cabeza y diversos huesos constituyen sus dos partes: el neurocráneo y viscerocráneo. El neurocráneo es la caja ósea del encéfalo y sus cubiertas membranosas. En un adulto, está formado por una serie de ocho huesos: cuatro impares centrados en la línea media (frontal, etmoides, esfenoides y occipital) y dos series de pares bilaterales (temporal y parietal). Los huesos del denominado neurocráneo en conjunto

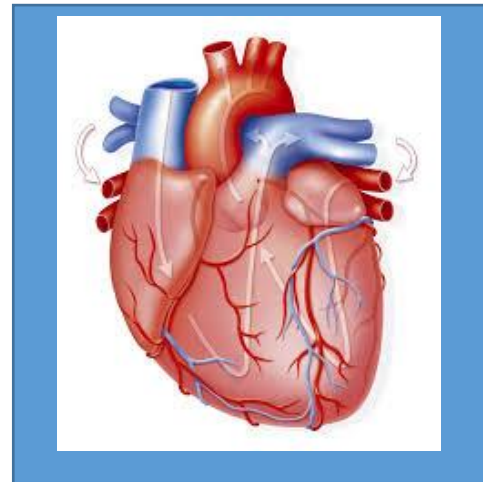


conforman otras dos estructuras anatómicas: Los huesos frontal, parietales y occipital suelen conformar una estructura de techo parecido a una cúpula, denominada calvaria o bóveda craneal, mientras que el hueso esfenoides y temporales forman parte de la base del cráneo. El espalncocráneo o viscerocráneo, también llamado esqueleto facial, constituye la parte anterior del cráneo y se compone de los huesos que rodean la boca (maxilares y mandíbula), la

nariz/cavidad nasal y la mayor parte de las cavidades orbitarias. Éste consta de 15 huesos irregulares: tres huesos impares centrados (mandíbula, etmoides y vómer) y seis huesos pares bilaterales (maxilar, cornete nasal inferior, cigomático, palatino, nasal y lagrimal). Los maxilares y la mandíbula albergan los dientes, o de otra forma dicho, proporcionan las cavidades y el hueso de sostén para los dientes maxilares y mandibulares. Los maxilares forman la mayor parte del esqueleto facial superior, fijado a la base del cráneo. La mandíbula forma el esqueleto facial inferior, siendo éste de carácter móvil al articularse con la base del cráneo en las articulaciones temporo-mandibulares.

Corazón

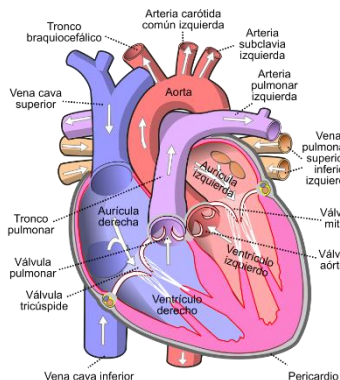
El corazón (del latín cor) es el órgano principal del aparato circulatorio. En los animales vertebrados, incluyendo el ser humano y mamíferos en general, es un órgano muscular hueco que funciona como una bomba aspirante e impelente que impulsa la sangre a través de las arterias para distribuirla por todo el cuerpo. El corazón humano tiene el tamaño de un puño, pesa entre 250 y 300 gramos en mujeres y entre 300 y 350 gramos en hombres, lo que equivale al 0,40 % del peso corporal.² Está situado en el centro de la cavidad torácica flanqueado a ambos lados por los pulmones.



El corazón es un órgano muscular hueco cuya función es bombear la sangre a través de los vasos sanguíneos del organismo. Se localiza en la región central del tórax, en el mediastino medio e inferior, entre los dos pulmones. Está rodeado por una membrana fibrosa gruesa llamada pericardio. El corazón tiene la forma de una pirámide inclinada, la porción puntiaguda de la pirámide está inclinada hacia la izquierda y abajo, mientras que la base mira hacia arriba y es el área de donde surgen los grandes vasos sanguíneos que llevan la sangre fuera del órgano. La parte inferior del corazón descansa sobre el diafragma mientras que las caras laterales están contiguas al pulmón derecho e izquierdo y la cara anterior se sitúa detrás del esternón.

El corazón está dividido en cuatro cámaras o cavidades: dos superiores, llamadas aurícula derecha (atrio derecho) y aurícula izquierda (atrio izquierdo); y dos inferiores, llamadas ventrículo derecho y ventrículo izquierdo. Las aurículas reciben la sangre del sistema venoso y la transfieren a los ventrículos, desde donde es impulsada a la circulación arterial.

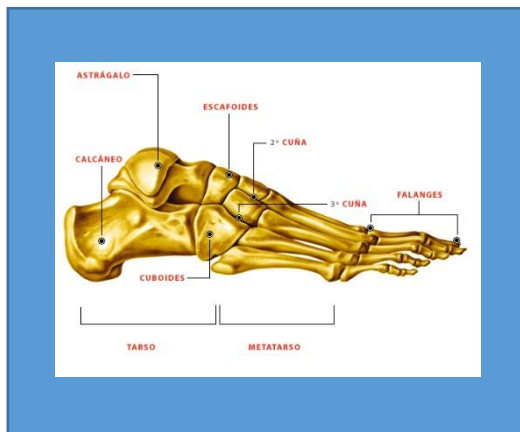
- Corazón derecho. La aurícula derecha y el ventrículo derecho forman el corazón derecho. La aurícula derecha recibe la sangre que proviene de todo el cuerpo a través de la vena cava superior y vena cava inferior. El ventrículo derecho impulsa la sangre no oxigenada hacia los pulmones a través de la arteria pulmonar.
- Corazón izquierdo. La aurícula izquierda y el ventrículo izquierdo forman el corazón izquierdo. Recibe la sangre oxigenada proveniente de los pulmones que desemboca a través de las cuatro venas pulmonares en la aurícula izquierda. El ventrículo izquierdo impulsa la sangre oxigenada a través de la arteria aorta para distribuirla por todo el organismo.



El tejido que separa el corazón derecho del izquierdo se denomina septo o tabique. Funcionalmente, se divide en dos partes no separadas: la superior o tabique interauricular, y la inferior o tabique interventricular. Este último es especialmente importante, ya que por él discurre el fascículo de His, que permite llevar el impulso eléctrico a las partes más bajas del corazón.

Pie

El pie es la porción terminal de una extremidad que lleva el peso del cuerpo y permite la locomoción. Es una estructura anatómica que se encuentra en muchos vertebrados. En muchos animales con pies, éste es un órgano independiente en la parte terminal de la pierna, y en general está compuesto por uno o más segmentos o huesos, incluyendo las zarpas, las pezuñas o las uñas. El ser humano usa sus pies para la locomoción bípeda, haciendo posible la posición vertical y la liberación de los miembros superiores. En su concepto, constituye una bisagra con el suelo, siendo crucial, pero como estructura inferior a menudo resulta infravalorado.



Los huesos que constituyen el pie:

Astrágalo. Es el único hueso del tarso que se articula con la pierna, quedando sujeto por la mortaja tibioperonea y articulándose caudalmente con el calcáneo y ventralmente con el escafoides.

Calcáneo. Tiene una forma irregularmente paralelepípeda representando su mitad posterior el talón. En su cara superior distinguimos dos carillas articulares para el astrágalo.

Escafoides. Presenta una forma navicular. Su cara posterior o proximal ofrece una excavación articular para el astrágalo. Su cara anterior o distal presenta tres facetas triangulares para articularse con las cuñas. En la parte interna del hueso se aprecia un saliente denominado tubérculo del escafoides y en la parte externa una carilla plana para el cuboides.

Cuñas o huesos cuneiformes. Son tres: primera o medial, segunda o intermedia y tercera o lateral. Todas presentan una cara proximal triangular articulada con el escafoides y una cara distal también triangular articulada con los cuatro primeros metatarsianos.

Cuboides. Tiene forma irregularmente cuboidea. Su cara proximal es lisa y se articula con el calcáneo. Su cara distal presenta dos facetas articulares para el cuarto y quinto metatarsiano.

Metatarsianos. Son pequeños huesos largos, que se disponen de dentro afuera con los nombres de primero, segundo, tercero, cuarto y quinto. No se encuentran en el mismo plano sino que forman un arco transversal, más elevado por dentro que por fuera.

Falanges. Se conocen con los nombres de primera o proximal, segunda o medial y tercera o distal o ungueal. El dedo gordo o hallux sólo tiene dos falanges: la proximal y la distal o ungueal. Son muy rudimentarias, presentando una base o extremidad proximal, una diáfisis muy corta y una cabeza o extremidad distal. Las superficies articulares de sus extremidades son trocleas rudimentarias.