72.45 - Proyecto Final Ingeniería Informática

Instituto Tecnológico de Buenos Aires

AirFitness

Generación de rutinas personalizadas con IA y corrección de ejercicios con ML

Autores:

Esteban Kramer (55148)

Martina Scomazzon (55410)

Tutor: Gonzalo Dolagaratz

Fecha: Diciembre 2020

Lugar: Ciudad Autónoma de Buenos Aires, Argentina

Índice

1. Justificación del Trabajo

AirFitness busca facilitar el proceso de realizar ejercicio físico y estar en forma sin la necesidad de ir a un gimnasio o tener un entrenador personal.

La elección de una rutina adecuada a las características y necesidades del individuo que desea entrenarse, representa un desafío sobre todo cuando este es principiante o tiene objetivos específicos.

Generalmente, el individuo comienza la búsqueda en internet donde la información resulta abrumadora y existen rutinas que prometen resultados que no siempre se cumplen.

A la hora de elegir una rutina, es importante que la misma se adapte a su nivel de experiencia, estilo de vida y que los ejercicios le resulten entretenidos y balanceados.

Una vez seleccionada la rutina se presenta un nuevo desafío, saber si los ejercicios que la conforman se realizan de forma correcta.

Con todo esto en mente, se propuso crear AirFitness. Utilizando inteligencia artificial, se generarán rutinas personalizadas para cada usuario y por medio de machine learning, se detectarán sus movimientos y verificará si los ejercicios se están realizando de forma correcta.

2. Estado del Arte

Antes de comenzar el proyecto se llevó a cabo una investigación con el fin de conocer el estado del arte. Se hizo foco en explorar de aplicaciones que realicen tareas similares a la problemática que este proyecto busca resolver.

Se encontró que no solo existen alternativas *mobile*, sino que también algunas se encuentran integradas a equipos deportivos, como pueden ser una bicicleta fija o una cinta. Otra alternativa para entrenar en el hogar son prototipos similares a un espejo o electrodoméstico en donde se muestra la rutina y como realizar los ejercicios.

A continuación se describirán cada una de alternativas evaluadas indicando cuál es su objetivo, cómo se presentan y si efectivamente lo cumplen.

2.1. VAY

VAY es una aplicación *mobile* que surge aproximadamente en el año 2018. Su objetivo es seguir el movimiento de las personas a través de una cámara con la finalidad que se desee. En otras palabras, es una API que busca digitalizar los movimientos del ser humano para luego, analizar su comportamiento. [VAY, 2020]

Esta aplicación puede ser utilizada para monitorear movimientos en tiempo real, de esta forma, garantiza un *feedback* sobre el movimiento que percibe de forma instantánea. Sus desarrolladores destacan que no es necesario el uso de sensores o equipo adicional para un correcto funcionamiento ya que con la cámara del dispositivo móvil es suficiente.

El uso de esta tecnología no está restringido al entrenamiento físico, sino que también puede ser aplicada en ámbitos de trabajo como por ejemplo, evitar accidentes laborales.

También ofrece una aplicación *mobile* conocida como "VAY Fitness Coach" la misma brinda una serie de rutinas predeterminadas, cada una con un "*coach*" asignado. Este, le indica al usuario qué ejercicios debe realizar y de ser necesario le corrige su postura e indica si lo está realizando a la velocidad correcta o no.

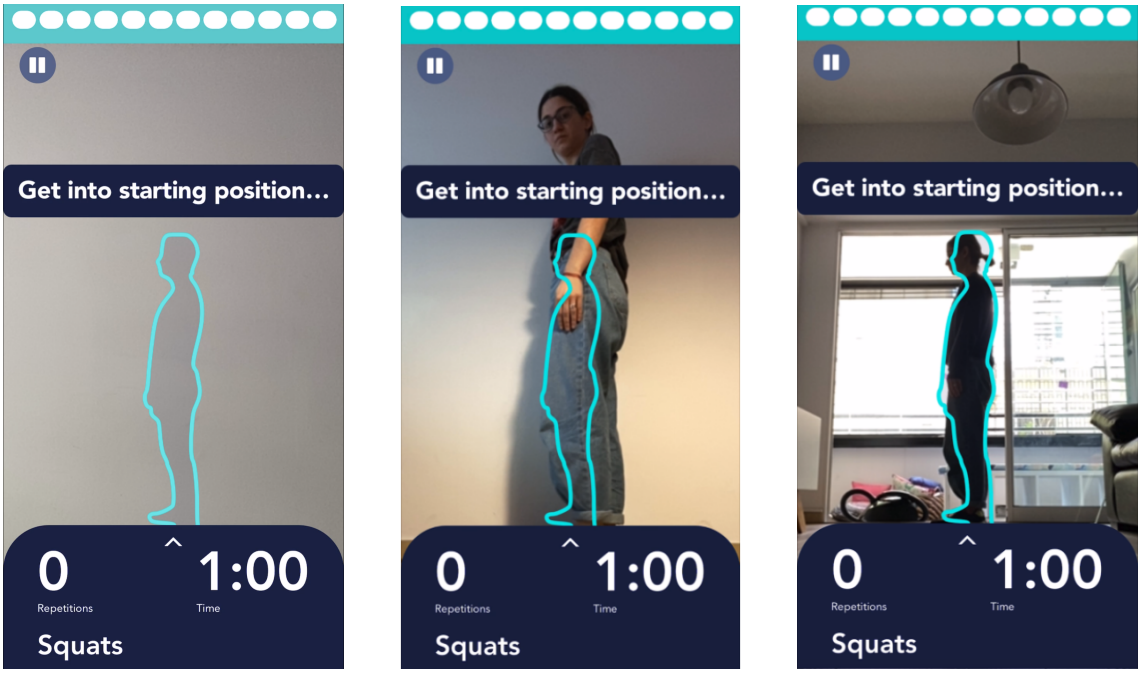
Con el objetivo de obtener un mayor conocimiento sobre esta tecnología se realizó un análisis de usabilidad investigando los datos de entrada necesarios y los procedimientos aplicados para obtener los resultados.

Para poder hacer uso de la aplicación es necesario inscribirse completando la siguiente información: nombre, apellido, altura y el objetivo de entrenamiento. Durante las pruebas realizadas se seleccionó como objetivo de entrenamiento la opción *bajar de peso*.

Para iniciar la rutina, la aplicación solicita al usuario que se coloque lo suficientemente lejos del dispositivo móvil de forma tal que su cuerpo quepa en la silueta mostrada en pantalla. Esta detección no es del todo precisa, por lo tanto, el usuario debe probar distintas configuraciones de distancia y altura antes de que su cuerpo sea detectado.

La primera limitación que presenta VAY Fitness Coach es que no permite visualizar la rutina completa. Al comenzar la rutina indica el ejercicio, muestra un vídeo de como se realiza y espera detectar el cuerpo para comenzar, si no lo logra, no hay forma de continuar con el entrenamiento.

Otra limitación que presenta es que no es posible modificar el objetivo de entrenamiento previamente ingresado en la inscripción del usuario. Ahora bien, si el usuario posee un mayor conocimiento sobre *fitness* puede seleccionar otro plan, y por consiguiente otro *coach*.



(a) **(b)** **(c)**

Figura 1: Interfaces de la aplicación VAY: (a) Silueta en la cual debemos colocarnos para que se detecte nuestro cuerpo. (b) Celular en el piso, distancia de 1.5 m, el cuerpo no fue detectado. (c) Celular en el piso, distancia de 1 m, el cuerpo fue detectado.

Una vez detectado el cuerpo, comienza el entrenamiento y una voz indica si esta realizando el ejercicio de la forma correcta o no. Cuando el ejercicio se realiza de forma incorrecta se informa pero la repetición se contabiliza, es decir, las repeticiones ejecutadas de forma incorrecta son contabilizadas en el total.

2.2. FitMe - AI Fitness Coach

FitMe es una aplicación *mobile* disponible solo para iOS y su objetivo es poder identificar los movimientos de un usuario en tiempo real para ayudarlo a mejorar su rendimiento a la hora de entrenar.
La misma identifica y provee *feedback* en tiempo real sobre cuan bien se esta realizando el ejercicio. Contabiliza de forma automática la cantidad de repeticiones y brinda un incentivo motivacional a lo largo de la rutina. [FitMe, 2019]

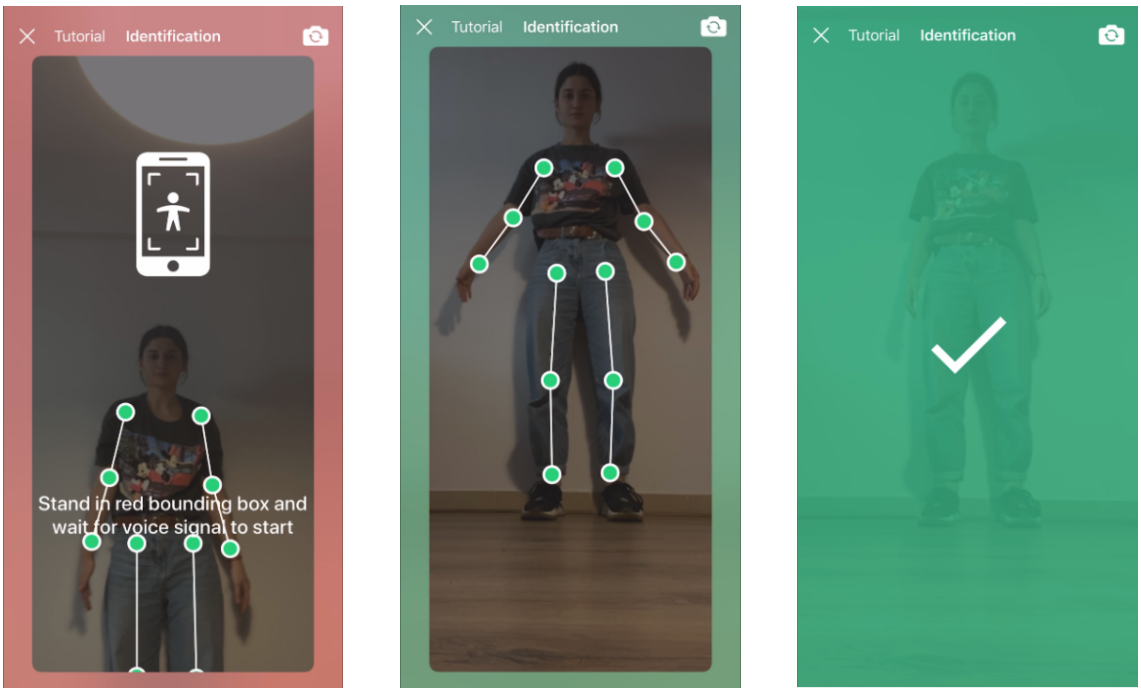
Al igual que VAY, existe una etapa de suscripción en donde el usuario debe crear una cuenta con información básica, frecuencia de entrenamiento, un objetivo y un nivel deportivo. En base a la información proporcionada, la aplicación brinda rutinas predeterminadas que se encuentra alineadas con sus características, incluyendo los días de descanso. Adicionalmente, la aplicación cuenta con un calendario en donde se pueden conocer todas las rutinas que hay por delante y ver el detalle de las mismas.

Por el contrario, a diferencia de VAY, FitMe permite acceder a sus rutinas, conocer los ejercicios que la componen y cuantas repeticiones se deben realizar de cada uno de ellos.

Si bien el usuario puede ver el detalle de la rutina, para poder completarla se requiere obligatoriamente el uso de la cámara para que los ejercicios sean detectados y contabilizados, caso contrario, la rutina se marca como no realizada. El mayor inconveniente que esto puede traer es que ninguna rutina sea considerada como completada.

Además de las rutinas, se pueden realizar ejercicios aislados. En dicho caso, el usuario tiene la posibilidad de indicar la cantidad de repeticiones que quiere lograr. Al llegar a este número, se dá por finalizado el ejercicio.

Al iniciar una rutina hay una breve introducción. Luego se le indica al usuario que coloque su dispositivo móvil en el piso, que tome una distancia suficiente tal que su cuerpo se vea en la pantalla y espere a que el borde de la misma se torne de color verde. Una vez logrado esto, comienzan a contabilizarse y corregirse los ejercicios.



(a) **(b)** **(c)**

Figura 2: Interfaces de la aplicación FitMe mostrando: (a) Introducción al ejercicio, espera detectar el cuerpo. (b) Identificó el cuerpo entero, coloca sus bordes en verde. (c) La aplicación ya podrá identificar y corregir el ejercicio.

Existen casos en los que la cámara logra identificar el cuerpo pero al iniciar el ejercicio no detecta los movimientos, imposibilitando la contabilización y corrección de los mismos.

La empresa que desarrolla *Fittonic*, tecnología utilizada por FitMe, aclara que la misma aún está en etapa de prueba por lo que los mensajes emitidos podrían no ser del todo correctos o aún más, no funcionar.

2.3. FitCam: Fitness Trainer

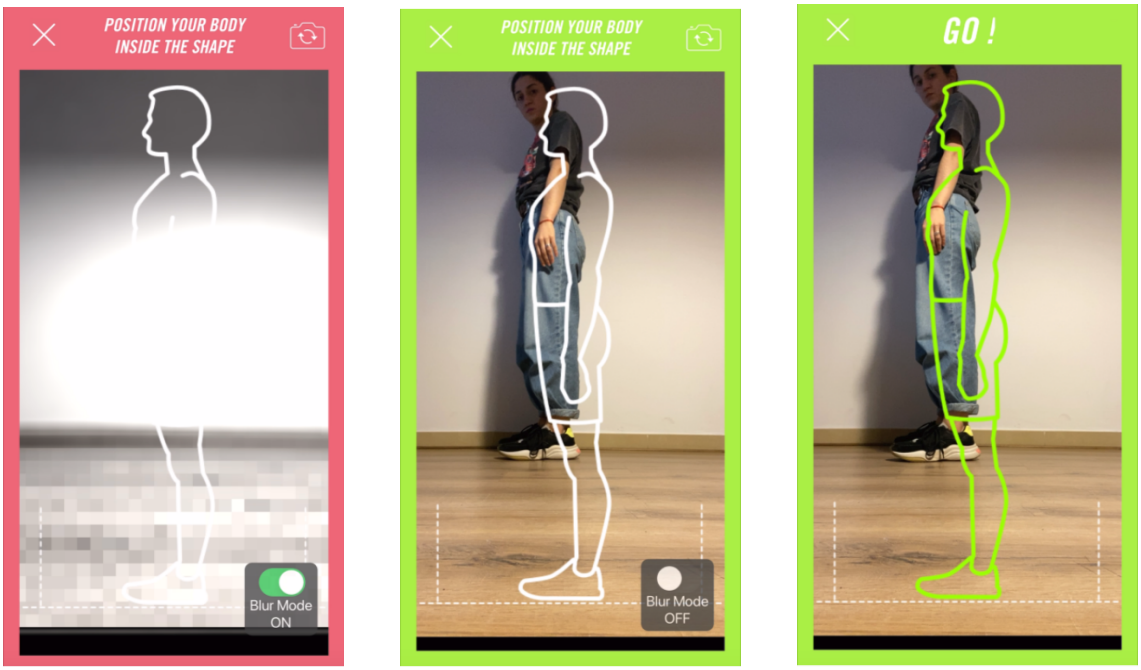
FitCam es una aplicación *mobile* para dispositivos iOS cuyo objetivo es la detección de postura y corrección de ejercicios luego de haber completado los mismos. Para monitorear e interpretar los movimientos de una persona la aplicación utiliza un SDK llamado FitCam Inside.[FitCam, 2019]

Dicha aplicación detecta cuatro ejercicios: sentadillas, flexiones de brazos, plancha alta y plancha baja. Estos fueron seleccionados bajo la justificación de que son suficientes para mantener todo el cuerpo activo.

Luego de descargar la aplicación, al igual que en todos los casos anteriores, el usuario debe crearse una cuenta con el objetivo de almacenar un registro sobre su actividad. En comparación con las otras aplicaciones, en este caso no se generan rutinas ni se solicita mayor información más que nombre y apellido.

Para llevar a cabo la corrección de un ejercicio el usuario debe seleccionar el ejercicio que desea realizar, esperar que la cámara lo detecte y luego, comenzar a realizarlo. La aplicación cuenta las repeticiones. Al presionar *finalizar* hace un análisis de los movimientos realizados arrojándole al usuario un resumen de la cantidad de repeticiones logradas junto con un puntaje. Este último, es calculado como el promedio del puntaje que el algoritmo le asignó a cada una de las repeticiones.

La detección funciona igual que en las otras aplicaciones, cuando el borde se torna de color verde significa que la misma está lista para comenzar a detectar los movimientos y contabilizar las repeticiones.

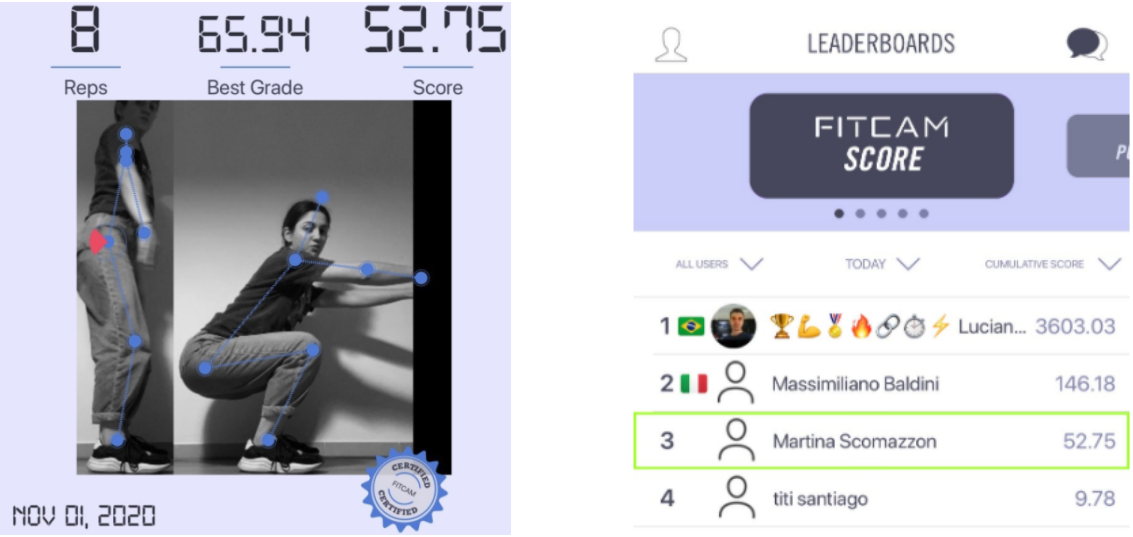


(a) **(b)** **(c)**

Figura 3: Interfaces de la aplicación FitCam mostrando: (a) Introducción al ejercicio, espera detectar el cuerpo. (b) Identificó el cuerpo entero, coloca sus bordes en verde. (c) La aplicación ya podrá identificar y corregir el ejercicio.

Como se mencionó anteriormente, en este caso la corrección no se realiza en tiempo real. Se hace un análisis de las repeticiones realizadas y se arroja un resultado. Si bien el objetivo de la aplicación es que el usuario se mantenga activo realizando actividad física, no proporciona mayores herramientas más allá de la corrección de cuatro ejercicios.

En este caso, el incentivo está en la competencia. Se puede competir con amigos o personas que utilicen la aplicación para ver quién realiza más repeticiones con el mejor puntaje posible, viéndose reflejado en un tablero de posiciones. A más ejercicios realizados, más puntos obtienen.



(a) **(b)**

Figura 4: Interfaces de la aplicación FitCam mostrando: (a) Imagen de análisis del ejercicio seleccionado: sentadillas. (b) Tabla de posiciones respecto a otras personas usando la aplicación.

En conclusión, la diferencia con las aplicaciones ya mencionadas es que en este caso se obtiene un puntaje promedio de las repeticiones realizadas. Además, evalúa el comportamiento del usuario luego de realizar el ejercicio y no durante.

2.4. Weight Lifting Plan: FitnessAI

Hasta ahora se han evaluado aplicaciones que tenían como principal objetivo la corrección de ejercicios. Ninguna permitía generar rutinas adaptadas al usuario, es decir, que se creen nuevas rutinas para cada individuo en función de sus necesidades y preferencias. Fitness AI lo hace. [FitnessAI, 2018]

Esta aplicación para dispositivos iOS utiliza un algoritmo en donde se crean rutinas específicas para cada usuario pero no posee la posibilidad de detectar movimiento y corregir ejercicios.

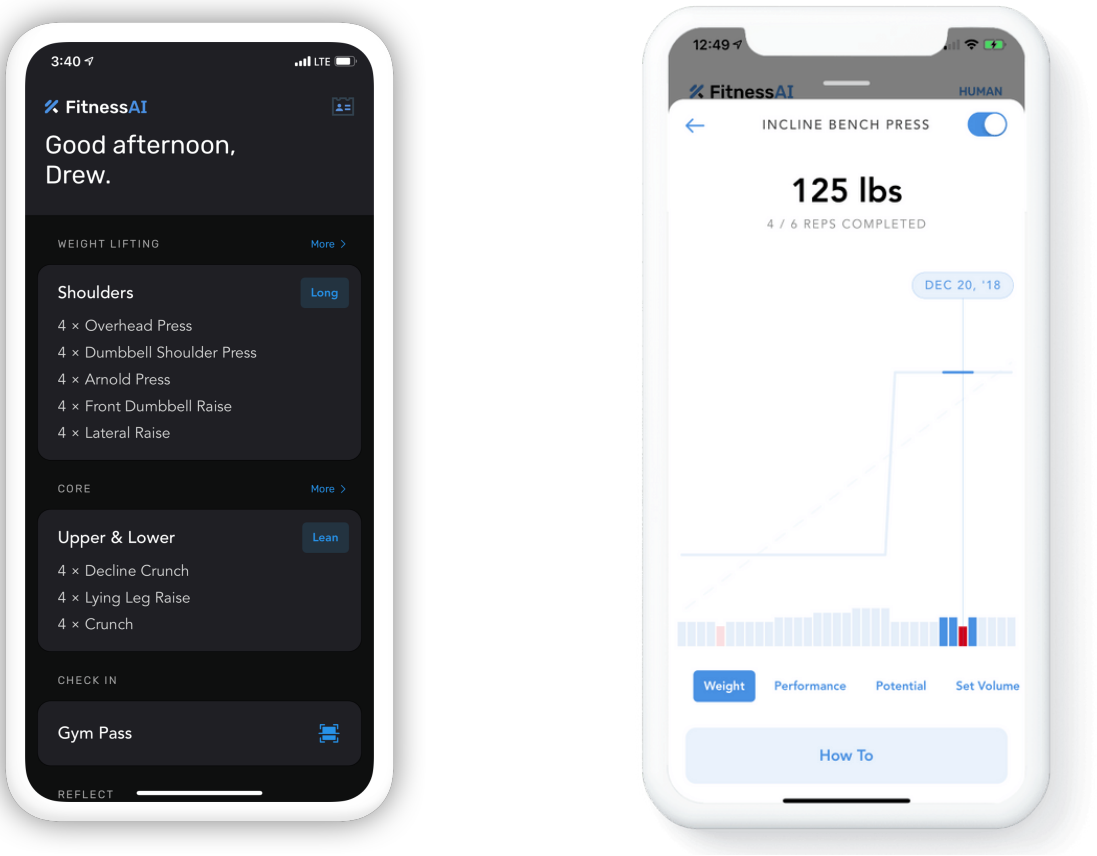
A lo largo del proceso de inscripción no solo se le pide al usuario la misma información básica mencionada previamente sino que además, se consulta qué elementos o equipamientos posee con el fin de buscar ejercicios adaptados a sus condiciones buscando maximizar los objetivos que este pudiera tener.

Una vez ingresado en la aplicación tendrá una rutina adaptada a sus necesidades. Además de mostrarle que ejercicios debe realizar para alcanzar su objetivo de entrenamiento, puede modificar las rutinas que se le han sido asignadas agregando más ejercicios, modificando el orden de los mismos o inclusive cambiando el equipamiento que tiene disponible y/o los músculos que desea ejercitar. Al realizar cualquiera de estos cambios, se le genera una nueva rutina adaptada a sus nuevos requerimientos.

Fitness AI es una aplicación que utiliza inteligencia artificial para generar rutinas personalizadas. Gracias a las más de 5.9 millones de rutinas con las cuales trabaja el algoritmo diseñado, optimiza los *sets*, repeticiones y peso necesario para cada ejercicio de la rutina. También cuenta con una serie de visualizaciones que le permite al usuario seguir su progreso.

Dentro de las configuraciones de la aplicación, es posible modificar la información básica, el equipamiento, la cantidad de veces a ejercitar por semana y si se desea tener más repeticiones o descansos más largos, ya que estos últimos, están definidos con un valor fijo para todos. Algo curioso es que nunca consulta al usuario cual es su nivel de experiencia entrenando o su estado físico.

Antes de comenzar la rutina se pueden ver cuales son los ejercicios que la componen, la cantidad de repeticiones que se deben hacer en cada *set* y una descripción detallada del mismo. En algunos casos se incluyen *tips* para aumentar o disminuir su dificultad.



(a) **(b)**

Figura 5: Interfaces de la aplicación FitnessAI mostrando: (a) Rutina generada para un usuario. (b) Comparación de progreso para *incline bench press*.

2.5. Peloton

Peloton es una empresa estadounidense que fabrica bicicletas fijas y cintas de caminar con una pantalla integrada, en donde el usuario puede reproducir clases, escuchar las indicaciones de un *coach* y seguir sus recomendaciones. [Peloton, 2012]

No posee inteligencia artificial ni corrección de ejercicios. Busca motivar a sus usuarios a través de más de mil clases de diversas disciplinas dictadas por diferentes profesores. La motivación se encuentra en la pantalla. En la misma se transmite la clase y si hay otros usuarios realizando la misma clase se puede interactuar con ellos.

Adicionalmente, obtiene información en tiempo real que usuario puede compartir con quien desee. Dicha información comprende desde el ritmo cardíaco hasta la resistencia.

Ofrece planes y desafíos por cumplir. Estos incluyen la posibilidad de competir contra otros usuarios o inclusive amigos. Por último, permite vincularlo a Spotify o Apple Music para tener la música que el usuario desee a la hora de entrenar.

En otras palabras, luego de investigar, quienes lo han probado lo describen como una clase de spinning en el hogar (en el caso de la bicicleta fija).

Por otro lado, la cinta fija incluye rutinas para realizar fuera de ella y de esta forma tener una clase completa de fuerza y cardiovascular en el hogar.

Recientemente, Peloton agregó a sus servicios una aplicación *mobile* que incluye más de mil clases a realizar sin ningún tipo de equipamiento.

En conclusión, Peloton ofrece tres versiones diferentes: integrado a la bicicleta fija, a una cinta fija o la más reciente, una aplicación *mobile*.

(a) **(b)**

Figura 6: Imágenes de Peloton: (a) Cinta fija con Peloton integrado. (b) Bicicleta fija con Peloton integrado.

2.6. Mirror

Se trata de una de las nuevas tendencias: mobiliario para realizar ejercicio en el hogar. Mirror es un espejo que se conecta mediante bluetooth al celular. [Mirror, 2020]

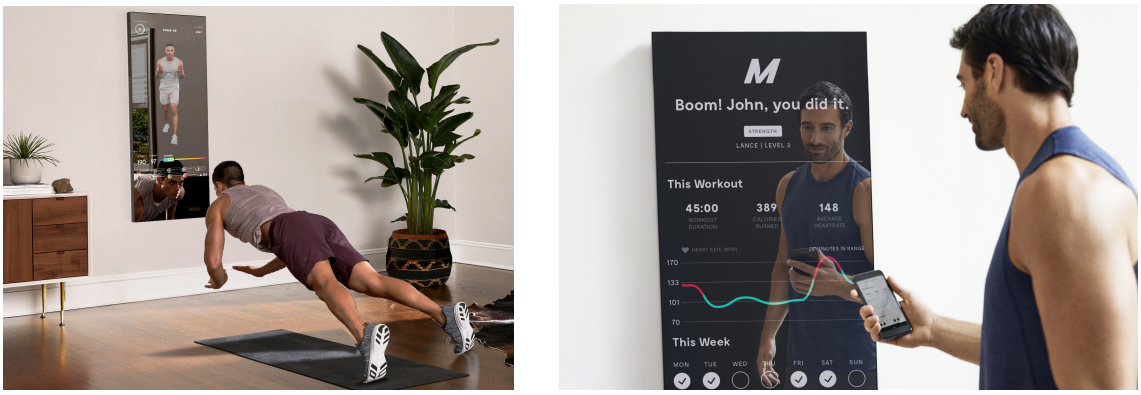
A través de una aplicación *mobile* el usuario debe seleccionar la rutina que desea llevar a cabo, la cual posteriormente se proyectará sobre el espejo.

Es decir, el espejo es una pantalla no táctil en donde se muestra un instructor realizando el ejercicio y el usuario debe repetir el movimiento frente al espejo para tener un registro propio de cuán bien lo está realizando.

Si bien su página web indica que busca optimizar el de entrenamiento del usuario en tiempo real, la rutina no es personalizada. Contiene una gran variedad de rutinas predefinidas que pueden elegirse de acuerdo al entrenamiento que se desee realizar, como por ejemplo: cardiovascular, yoga, fuerza, etc. También cuenta con clases en vivo en donde el instructor puede ver al usuario y dar un *feedback* en tiempo real. Esto no sucede con las clases predefinidas.

Dado que el producto no se comercializa en el país la única forma de conocer más sobre él fue a través de reseñas en YouTube. Las opiniones al respecto son similares entre ellas: se destaca el alto costo que implica comprarlo, U$D 1500 con un extra mensual para poder acceder a las clases predefinidas. Esto incomoda a los usuarios ya que no encuentran la diferencia con seguir vídeos de YouTube o cualquier otra plataforma frente a un espejo. Aún así, consideran que posee buenas clases y pueden ser altamente efectivas.

Al finalizar la clase, el espejo le muestra al usuario un resumen de la rutina que ha llevada a cabo con una infografía sobre su ritmo cardíaco y cuántas calorías quemó.



(a) **(b)**

Figura 7: Imágenes de Mirror: (a) Usuario realizando clase con Mirror. (b) Usuario observando resultados en Mirror.

2.7. Tempo

Otra alternativa es Tempo. Este se presenta como el nuevo y más completo gimnasio en casa compuesto por sensores que analizan los movimientos del usuario, corrigiendo su postura y contando repeticiones. También cuenta con un pequeño compartimento que contiene discos con diferentes pesos. Los mismos pueden ser utilizado junto con una barra ubicada en la parte posterior del equipo para realizar los ejercicios indicados en pantalla.[Tempo, 2018]

Con la incorporación de sensores Tempo genera un modelo 3D del usuario, capturando cada matriz de este con la finalidad de detectar su postura y conocer cuán buena es. En caso de estar haciéndolo mal, corregirlo y de esa forma evitar lesiones. De este modo se obtiene un *feedback* en tiempo real de como mejorar, perfeccionar la forma y contar las repeticiones bien realizadas del ejercicio. Para respetar la privacidad del usuario el modelo se genera solo a partir del cuerpo del usuario y no de su entorno.

Además, recomienda cuanto peso debe utilizar el usuario para la realización del ejercicio. Puede hacer esto ya que posee un registro completo del peso utilizado en el pasado y sabe cuando aumentarlo. Por último, le recomienda al usuario que clases debería realizar para poder cumplir con sus objetivos. No se generan rutinas de forma personalizada para el usuario, pero si, le recomienda actividades que puedan ayudarlo a alcanzarlos.

Una vez más, como el producto no se comercializa en el país se recurrió a vídeos y reseñas del mismo para poder conocer que tan bien funciona. En este caso el precio es más elevado llegando a los U$D 2000 junto con una suscripción mensual que incluye más funcionalidades.

A diferencia del caso anterior, el elemento no es un espejo, sino más bien un electrodoméstico. La pantalla es táctil y se maneja todo desde el equipo mismo y no es necesaria una conexión adicional al celular.

Existen varias propuestas de entrenamiento que varían entre 6 a 12 semanas y se pueden elegir rutinas pre-diseñadas o realizar una clase grupal en vivo. Si bien las rutinas no son personalizadas, Tempo genera recomendaciones según las actividades previamente elegidas por el usuario.

Tempo no solo incluye las pesas sino que también, trae consigo una banda para monitorear el ritmo cardíaco del usuario durante la rutina. A través de ella, no solo logrará tener un registro y análisis de su corazón sino que además, calcula la cantidad de calorías quemadas. Al finalizar la actividad mostrará un resumen de su desempeño.

Los sensores corrigen la postura y cuentan repeticiones, cabe aclarar debido a que se trata de un producto nuevo no lo puede hacer con todos los ejercicios, fundamentalmente con aquellos que son en el piso, por ejemplo, abdominales. En dichos casos, Tempo aclara que aún no ha logrado recabar la suficiente cantidad de información para hacer uso de la inteligencia artificial y determinar la correcta postura del ejercicio.

Por último, para motivar a los usuarios el sistema posee una estrategia similar a la ya vista en *FitCam*: una tabla de posiciones. En función de la cantidad de repeticiones correctas realizadas, el tiempo y las métricas cardíacas, el usuario obtiene un puntaje y posiciona en el *ranking* junto a otras personas que realizaron la misma rutina.

(a) **(b)**

Figura 8: Imágenes de Tempo: (a) Usuario realizando clase con Tempo. (b) Equipamiento Tempo.

3. Objetivos

Luego de identificar cuál era el estado del arte restan definir los objetivos del proyecto. Dentro del análisis anterior se detectaron dos problemáticas. En primer lugar, ninguna de las aplicaciones o tecnologías evaluadas combinan la corrección de ejercicios junto con la generación de rutinas 100 % personalizadas (en función de las necesidades y configuraciones del usuario). En segundo lugar, existen casos en los que la corrección de postura frente a los ejercicios necesita de sensores especiales.

El proyecto busca atacar esta problemática e incluir todo en una única aplicación: la corrección de ejercicios junto con la generación de rutinas personalizadas, un *personal trainer* en el celular. Y con la ayuda de la cámara de dispositivos móviles, como el celular o *tablet* y sin ningún equipamiento extra, los usuarios puedan corroborar o conocer cuan bien están realizando ciertos ejercicios, los cuales de ser mal realizados podrían ocasionarles una lesión.

Por otro lado, busca generar rutinas personalizadas, es decir, se arman en función de las preferencias y configuraciones del usuario. Si el usuario realiza algún cambio sobre sus configuraciones, estas rutinas se adaptarán a dicho cambio.

Además, con el pasar del tiempo el usuario irá subiendo de nivel, haciendo que cada vez se le presenten ejercicios más difíciles o con mayores repeticiones.

Dentro de las preferencias o configuraciones, el usuario podrá especificar a que tipo de elementos tiene acceso, ya sea equipamiento deportivo o maquinas de gimnasio, cuantas veces por semana va a entrenar e inclusive cual cree que es su nivel de experiencia, y el objetivo que busca lograr con la aplicación: bajar de peso, definir o mantenerse en forma.

Existen usuarios que prefieren tener una rutina enfocada a deportes. Para ellos se brinda la alternativa de elegir uno en particular, las opciones son: tenis, fútbol y hockey. Al terminar la cantidad de rutinas indicadas para dicha semana, automáticamente, se generan nuevas rutinas. En conclusión, el objetivo es poder desarrollar una aplicación *mobile* que incluya todo lo planteado y de esta forma lograr que cualquier individuo que disponga de acceso a un dispositivo móvil tenga acceso a un profesional del deporte.

4. Requerimientos

A continuación se detallan los requisitos tanto funcionales como no funcionales junto con una breve descripción de cada uno de ellos.

4.1. Funcionales

4.1.1. Registrar usuarios

El registro de usuarios no solo permite conocer quienes son los que usan la aplicación, sino que además otorga la posibilidad de generar rutinas totalmente personalizadas para cada individuo. Por otra parte, le brinda al usuario la oportunidad de acceder a sus rutinas desde cualquier dispositivo y en todo momento.

A la hora de registrarse se le pedirá la siguiente información básica: nombre, apellido, edad, peso y sexo. También se le consultará sobre sus preferencias en cuanto a los objetivos de entrenamiento o si desea realizar una rutina diseñada para un deporte específico, el equipamiento disponible y su nivel de experiencia entrenando.

De esta manera **cada usuario** tiene que registrar los siguientes datos:

- Nombre
- Apellido
- Email
- Año de nacimiento
- Sexo
- Altura
- Peso
- Objetivo de entrenamiento
- Deporte que practica
- Frecuencia
- Nivel
- Puntos

4.1.2. Login de usuarios

Continuando con la misma lógica, al permitirle al usuario hacer *log in* desde cualquier dispositivo móvil podrá ver las rutinas que le han sido asignadas.

4.1.3. Cargar y ver ejercicios

Cada usuario tendrá una rutina asignada en función de la cantidad de días que se comprometió a entrenar, disponibilidad de equipamiento y nivel de entrenamiento. Una vez que accede a la aplicación el usuario podrá ver todas sus rutinas e ingresar a ellas para ver el detalle: ejercicios y cantidad de repeticiones. Incluso puede acceder a la descripción de un ejercicio en particular, donde encontrará indicaciones de como realizarlo.

En resumen, **cada ejercicio** va a contar con:

- Nombre
- Músculos que ejercita
- Vídeo (opcional)
- Dificultad
- Instrucciones
- Equipamiento necesario
- Imágenes

4.1.4. Generación de rutinas

Junto con las preferencias del usuario y sus configuraciones se generarán rutinas con inteligencia artificial, siendo estas 100 % personalizadas.

Cada rutina personalizada cuenta con:

- Listado de ejercicios a realizar.
- Un indicador que permite saber si la rutina fue finalizada o no.
- Usuario, al que está vinculada la misma.

4.1.5. Generación de rutinas por defecto

Si bien cada usuario tiene sus rutinas personalizadas puede suceder que el mismo quiera realizar una rutina adicional por fuera de su plan. Por dicha razón se decidió incorporar la posibilidad de tener rutinas genéricas, es decir, rutinas que van a ser iguales para todos los usuarios. Dichas rutinas genéricas podrán ser cargadas por el administrador del sistema.

Cada rutina genérica tendrá:

- Listado de ejercicios.

Todas las rutinas van a poder ser marcadas como realizadas para que el usuario avance de nivel.

4.1.6. Evolución de niveles

A medida que el usuario avanza en su plan personalizado, incrementa su nivel, asignándole ejercicios cada vez más difíciles. Para ello se definieron tres niveles:

- Principiante: Ejercicios de movimientos simples y pocas repeticiones.
- Intermedio: Ejercicios de complejidad media/avanzada con una cantidad intermedia de repeticiones.
- Avanzado: Ejercicios de complejidad media/avanzada con muchas repeticiones.

Luego de 30 rutinas realizadas, el usuario pasa de principiante a intermedio y luego de 50 rutinas más, a avanzado.

4.1.7. Cambio de equipamiento

El usuario podrá cambiar el equipamiento con el que cuenta para hacer ejercicio y generar nuevas rutinas al instante.

4.1.8. Corrección de postura de ejercicios

Por último, el usuario tiene la posibilidad de utilizar la cámara del dispositivo móvil para que la aplicación reconozca sus movimientos y en caso de estar haciéndolo mal, corregirlo. Si el movimiento fue realizado correctamente, suma una repetición.

4.2. No Funcionales

A partir de los requisitos definidos anteriormente, los atributos de calidad priorizados fueron los siguientes:

- **Disponibilidad**:
  - El sistema debe estar disponible cuando el usuario lo requiera.
- **Usabilidad**
  - Facilidad de aprendizaje.
  - Facilidad de uso de la plataforma.
  - Estética de la interfaz de usuario.
- **Escalabilidad**
  - El sistema debe poder funcionar correctamente a medida que su base de usuarios vaya creciendo.
- **Confiabilidad**
  - El sistema debe ser tolerante a fallos.
  - Debe funcionar con conexiones inestables.
- **Mantenibilidad**
  - El sistema debe estar diseñado para agregar nuevas funcionalidades con facilidad.

5. Análisis

A lo largo de esta sección se darán a conocer cada uno de los elementos considerados, la investigación llevada a cabo y cual fue la solución al respecto.

5.1. Usuarios Representativos

Antes de comenzar con el desarrollo de la aplicación y las decisiones tanto de usabilidad como de implementación, es necesario definir quienes usarán la aplicación, bajo qué condiciones y fundamentalmente cómo y cuándo.

Este proceso se dividió en tres etapas. En una primera etapa, se generaron hipótesis evaluando quienes podrían ser los usuarios representativos y qué características tendrían. En la segunda etapa se buscaron individuos que estén dispuestos a contestar una breve encuesta. En la última etapa, se evaluaron la veracidad de dichas hipótesis.

Las características principales a ser evaluadas fueron:

- Edad.
- Conocimiento tecnológico.
- Nivel deportivo.
- ¿Utilizó alguna vez una aplicación deportiva?

La edad permite segmentar la población entre los usuarios y nivel tecnológico para poder comprender que consideraciones se debían tener en cuenta en la usabilidad de la aplicación.

Con la finalidad de conocer como se deben pensar e implementar los niveles deportivos es importante conocer el nivel de los usuarios.

Por último, saber si el usuario utilizó o no una aplicación deportiva anteriormente brinda la seguridad de conocer las ventajas y desventajas de las mismas.

Se definieron como usuarios representativos personas de entre 20 y 30 años con un conocimiento medio-alto de tecnología y con facilidad para utilizar un dispositivo móvil. Aquellos dispuestos a utilizar la aplicación son usuarios que ya han utilizado una aplicación de entrenamiento anteriormente. El nivel deportivo puede ser variable.

En base a los usuarios definidos, se buscaron potenciales clientes que respondan una breve entrevista. Dado que se buscaba expandir el conocimiento sobre los ítems mencionados anteriormente, se desarrolló una entrevista que no tenía preguntas guiadas. El objetivo de la misma era determinar si el usuario representativo hipotético era correcto o no.

Se entrevistó un grupo de personas entre 20 y 60 años con conocimiento tecnológico variado, es decir, quienes solo saben usar aplicaciones de mensajería instantánea y redes sociales e individuos que poseen un mayor manejo de su dispositivo móvil. Por último, no se consideró relevante evaluar el nivel deportivo de las personas a entrevistar.

Al analizar los resultados, efectivamente quienes estarían de acuerdo en utilizar la aplicación son personas que han usado una aplicación deportiva con anterioridad, fundamentalmente porque comprenden cómo se utilizan y además, conocen las debilidades que estas pueden tener.

Por otro lado, no se hizo una pregunta específica sobre cuál era el nivel deportivo de las personas, sino que a partir de determinadas preguntas como lo pueden ser: *¿Cuántas veces por semana entrena?, ¿Entrena con peso?*, logramos deducir el nivel deportivo aproximado del individuo. A partir de esto se determinó que no existe una relación estrecha entre el nivel deportivo y la voluntad de usar la aplicación.

Si bien existen individuos entre 50-60 años dispuestos a utilizar la tecnología, son casos aislados que no pertenecen a la media. Quienes más contestaron que usarían la aplicación son personas de entre 20-40 años, lo que determina que se debe ampliar el rango etario en 10 años.

En conclusión, los usuarios representativos serán individuos entre 20 y 40 años con un conocimiento de tecnología medio-alto, de diferentes niveles deportivos y con conocimiento previo o utilización de aplicaciones deportivas.

Las entrevistas no solo brindaron conocimiento sobre quienes serían los usuarios representativos sino que también sobre lo que espera el usuario de la aplicación y que consideraciones se deben tener a la hora de diseñar e implementar el proyecto.

5.2. Análisis de Plataforma

Si bien a lo largo de todo el informe se hace mención a una aplicación *mobile*, cabe destacar que no fue la primera elección. Antes de comenzar el desarrollo de la misma se consideraron diferentes alternativas.

Se plantearon tres posibilidades: una aplicación de escritorio, una aplicación web y por último, una aplicación *mobile*.

La primera alternativa, una aplicación de escritorio, fue rápidamente descartada pues se desea que la aplicación sea fácilmente transportable, permitiéndole el usuario disfrutar de ella en el gimnasio, la plaza o en su casa.

Impulsados por la voluntad de que la aplicación sea fácilmente transportable, se determinó que era de suma importancia su facilidad de uso en dispositivos móvil.

Hoy en día las personas se trasladan casi de forma involuntaria con un dispositivo móvil, pero no así con una computadora, aún más, hay personas que han sustituido el uso de la computadora por un dispositivo móvil, fundamentalmente las personas mayores.

Por lo tanto, las alternativas restantes a evaluar son dos: una aplicación web *responsive*, o una aplicación *mobile*. Debido a las razones previamente mencionadas, se decidió desarrollar una aplicación *mobile*.

Insistiendo en que el objetivo era llegar a la mayor cantidad de usuarios posibles y el tiempo de desarrollo era limitado, se utilizó React Native. De esta forma un mismo código logra abarcar mayor cantidad de usuarios posibles, caso contrario se debería haber desarrollado la misma aplicación en dos lenguajes diferentes.

5.3. Tecnologías

Conociendo a los usuarios representativos y teniendo en claro que se desarrollaría una aplicación *mobile* con React Native, era necesario encontrar la tecnología que brinde la capacidad de lograr el objetivo de estimación de pose. Para ello se evaluaron dos alternativas: PoseNet y OpenPose.

A continuación, una breve introducción de qué trata cada una, las ventajas y desventajas, su comportamiento y porque se decidió trabajar con una de ellas en particular.

5.3.1. PoseNet

PoseNet es un modelo de visión que se puede utilizar para estimar la pose de una persona en una imagen o vídeo estimando dónde están las articulaciones claves del cuerpo. Es importante aclarar que la tecnología no reconoce quien está en la imagen, sino que aproxima a donde se deberían encontrar los puntos de las articulaciones del individuo. [PoseNet, 2020]

Cada uno de los 16 puntos claves está indexado con un ID, un valor de confianza entre 0.0 y 1.0, siendo 1.0 el valor más alto. Este valor representa el nivel de exactitud con el cual se está aproximando la posición de dicho punto.

5.3.2. OpenPose

A través de un modelo matemático desarrollado en Carnegie Mellon University se puede detectar en tiempo real el movimiento de una o múltiples personas. Este, detecta más de 135 puntos claves que le permite estimar las articulaciones del ser humano frente a la cámara. La tecnología se encuentra disponible en Python y C ++.

Ambas tecnologías son de código abierto y se pueden encontrar fácilmente en GitHub. Los dos buscan el mismo objetivo, detectar en tiempo real los movimientos de un ser humano.

En un estudio[suprnrdy, 2020] donde se analizan y comparan ambas tecnologías, se demuestra que la detección de poses por parte de OpenPose es superior a PoseNet.[OpenPose, 2018]

Si bien PoseNet procesa rápido la información sucede que existen varias poses que no son capturadas, generando una intermitencia de data y logrando que por momentos no se capture el verdadero movimiento.

El problema de OpenPose es que no está disponible para React Native, imposibilitando el trabajo previamente diseñado.

PoseNet posee mayor documentación, más cantidad de individuos utilizándolo y por ende más información disponible en internet, facilitando la búsqueda de información en caso de ser necesario.

En función de lo planteado, se decidió utilizar PosNet para la detección y posterior corrección de los ejercicios

5.4. Rutinas

5.4.1. Selección de Algoritmo

Durante el proceso de investigación para la selección del algoritmo se consultaron diversas fuentes. Aún existiendo dudas sobre como abordar el problema, se contactó al Ingeniero Rodrigo Ramele, profesor del Instituto Tecnológico de Buenos Aires, quien brindó una gran ayuda sobre cómo abordar el problema.

El problema a resolver era uno de optimización, donde sobre la base de la información del usuario se debía obtener un subconjunto de ejercicios que se adapten a él. A partir de sus restricciones y variables se requiere una función de costo que regule la asignación de ejercicios, buscando minimizarla. En otras palabras, son problemas que buscan el mejor elemento de un conjunto de ítems y pueden ser resueltos por un algoritmo de ordenación o una meta-heurística.

Dentro de los algoritmos de optimización más conocidos se encuentran: Greedy, Linear Programming, Simulated Annealing, Tabu Search, GRISP y Deep Learning. Dentro de la investigación, se agregó un sexto, BAT.

BAT es un algoritmo de meta-heurística de optimización desarrollado por Yang en el año 2010. Su objetivo era generar un algoritmo simple, eficiente y aplicable a diferentes problemas. Está inspirado en el fenómeno de eco-localización de murciélagos. [Yang, 2010]

Para el desarrollo del mismo se plantean tres características fundamentales del comportamiento de los murciélagos:

1. Todos los murciélagos utilizan geo-localización para identificar su distancia a la presa.
2. Los murciélagos vuelan de forma aleatoria a una velocidad v_i en la posición x_i (solución) con una frecuencia variable $Q_i \epsilon [Q_{min}, Q_{max}]$ con un pulso de emisión de $r_i \epsilon [0, 1]$ y un ruido de $A_i \epsilon [A_0, A_{min}]$ donde A_{min} es una constante.
3. Tanto el ruido como la emisión, son configurables según el objetivo.

Cabe aclarar que la búsqueda está definida por un camino aleatorio local. La selección del mejor destino continúa hasta alcanzar ciertos criterios de parada. Para lograr esto, se ajusta la frecuencia con el objetivo de controlar el comportamiento dinámico de un conjunto de murciélagos, alcanzando un equilibrio entre explotación y exploración.

Se investigó sobre dos algoritmos más: Simulated Annealing y Greedy, meta-heurística que a través de dos enfoques diferentes buscan resolver un mismo problema de optimización.

Simulated Annealing es un algoritmo enfocado en técnicas probabilísticas donde cada iteración del sistema tiene asociado un estado y un conjunto de vecinos. En cada iteración, se decide si avanzar de estado a alguno de los vecinos ó permanecer en el mismo. Estas iteraciones y movimientos entre vecinos alteran el sistema, buscando la solución cuya función de costo se minimice. Se repite el proceso hasta que el sistema alcance un estado que sea lo suficientemente bueno para cumplir con el objetivo ó la cantidad de iteraciones permitidas sea alcanzada.

Por el contrario, **Greedy**, en cada iteración elige la opción que más se adecue a la solución del problema, buscando la opción óptima. En otras palabras, toma todos los datos de un problema y en cada iteración, bajo ciertas reglas, elige cual es el elemento que puede ser agregado a la solución final. Por lo tanto, la solución brindada por el algoritmo es la que se crea en función de la suma de las elecciones realizadas en cada iteración. En cada paso, la decisión se realiza en función de lo ya seleccionado o agregado a la solución final pero no en lo que puede venir, es decir, se podrían alcanzar mínimos locales dificultando la resolución del problema.

Cabe destacar que los tres logran cumplir con el objetivo, pero dado que estamos frente a un trabajo experimental y contamos con tiempo limitado, se eligió profundizar en dos de ellos: Greedy y Simulated Annealing, algoritmos que han sido estudiados a lo largo de la carrera y que el Ingeniero Rodrigo Ramele recomendó. Se profundizará sobre ellos en la sección de implementación.

5.4.2. Reglas

Junto con la elección del algoritmo a ser utilizado se definieron reglas básicas que fueron mejoradas a través de una serie de pruebas empíricas. Estas serán utilizadas para componer la función de costo que el algoritmo buscará minimizar.

Al igual que el caso anterior, se profundizará sobre el desarrollo de las reglas utilizadas en los algoritmos generadores de rutina en la sección de implementación. Si bien las reglas han sido modificadas, existen reglas básicas que fueron pensadas desde un inicio. Por ejemplo, el orden de los ejercicios que debería tener una rutina.

Se entrevistaron preparadores físicos[1] quienes explicaron que en caso de entrenamientos de cuerpo completo se pueden organizar las rutinas por partes: primero piernas, luego brazos y por último abdominales o intercalado, siempre teniendo en cuenta que no puede haber dos ejercicios categorizados como difíciles o de alta intensidad de forma consecutiva. El orden de los ejercicios debe ser progresivo, comenzar con aquellos de intensidad baja y finalizar con los más intensos.

5.4.3. Ejercicios

La elección de los ejercicios incluidos en la aplicación y la información solicitada al usuario fue definida con profesores de educación física experimentados en el área. Algunos más jóvenes, quienes brindan una nueva mirada sobre el entrenamiento personal y otros que han estado más tiempo en el área.

Para la etapa de corrección de postura se seleccionaron aquellos ejercicios que abarquen la mayor cantidad de músculos posibles, de forma tal que luego de varias repeticiones el usuario pueda ejercitar todo el cuerpo. Los elegidos fueron: *squats, bicep curl, deadlift, lateral kick, lateral raise, scarecrow extensions, tricep extensions* y *front raise*. Según los profesionales las personas suelen realizarlos de forma incorrecta.

Con la ayuda de profesionales, análisis de imágenes e información encontrada en internet, se definieron los ángulos que deben conformar cada una de las articulaciones involucradas en los ejercicios escogidos para que los mismos sean realizados correctamente.

[1] Loana Vietta, Manuela Serra, Pablo Ortiz

5.5. Registro de usuarios

Los datos que son de relevancia para la generación de las rutinas personalizadas fueron evaluados y consultados con profesionales del área. Algunos han sido considerados y otros serán evaluados en una próxima iteración de la aplicación.

Los elementos evaluados como importantes a la hora de generar una rutina son:

- Edad
- Género
- Peso
- Lesiones o dolores
- Practica deporte, o no
- Objetivo de entrenamiento
- Si posee equipamiento deportivo

El género es importante preguntarlo ya que existen ejercicios que las mujeres consideran que son difíciles de realizar, o que se deberían evitar a lo largo del período menstrual porque podrían dañar el suelo pélvico.

6. Diseño

6.1. Front End

A la hora de elegir la tecnología a utilizar se evaluó la posibilidad de desarrollar la aplicación en código nativo: Java, Swift ó React Native.

React Native es un framework desarrollado por Facebook que permite crear aplicaciones nativas para iOS y Android utilizando el mismo código.

Para tomar la decisión se tuvieron en cuenta varios factores:

- Compatibilidad con TensorFlow/PoseNet
- Experiencia con la tecnología
- Performance

Todas las tecnologías mencionadas anteriormente cuentan con librerías de TensorFlow, por lo tanto, dicho factor no impactó en la decisión final.

En segundo lugar, no contamos con experiencia desarrollando aplicaciones en Swift y con poca experiencia desarrollando aplicaciones para Android en Java.

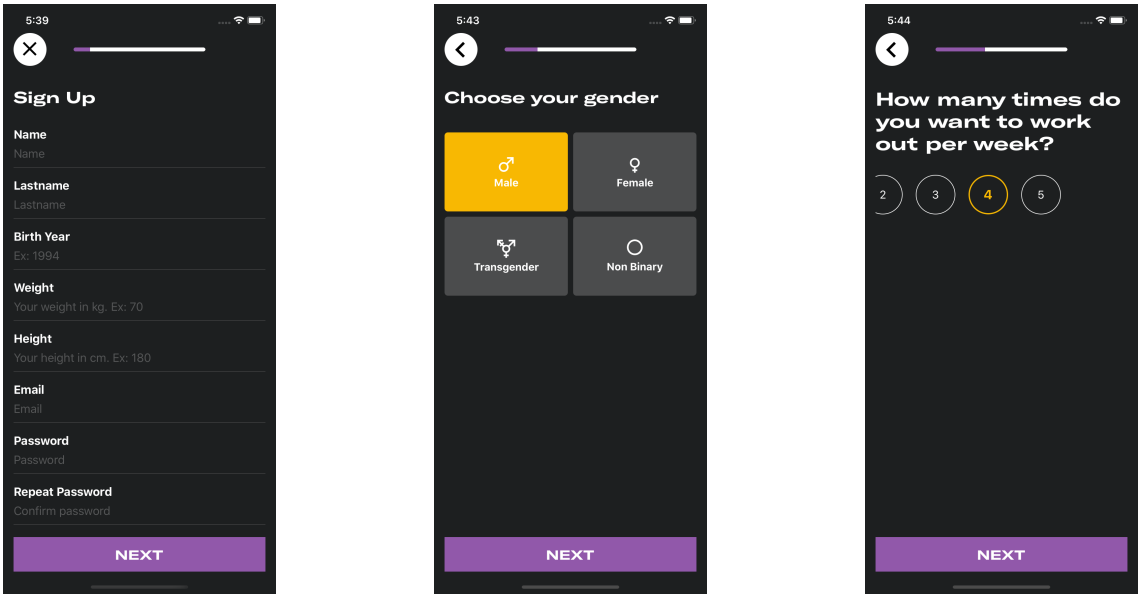
Dado que no se conocía cuan performante podría ser una aplicación desarrollada en React Native para la detección de movimientos en tiempo real, se decidió hacer un prototipo de prueba en Swift y otro en React Native.

Se pudo comprobar que si bien la aplicación en Swift funciona más rápido, la de React Native funciona correctamente y permite llegar a más usuarios.

Por lo mencionado anteriormente, se decidió desarrollar la aplicación en React Native.

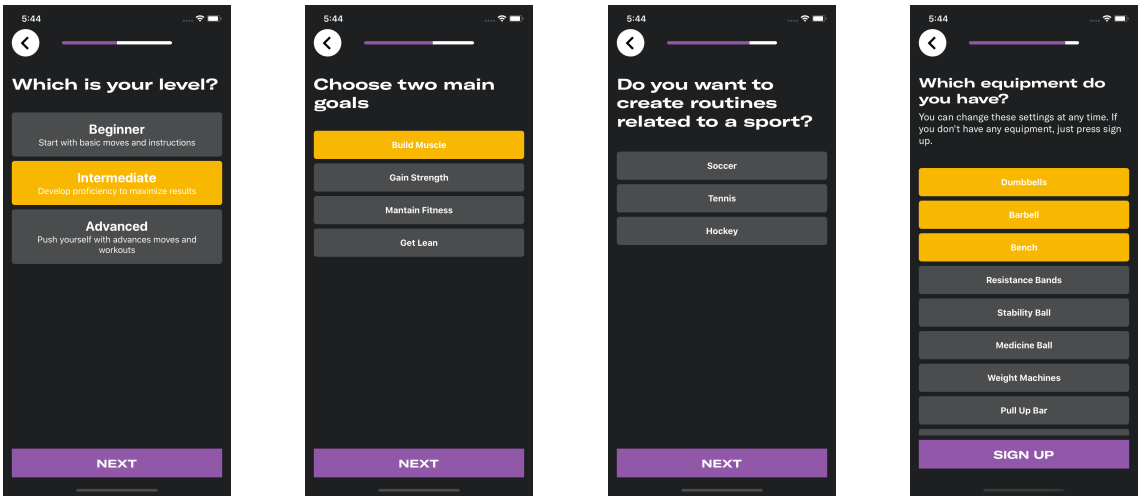
6.1.1. Vistas

Registro de un nuevo usuario



(a) **(b)** **(c)**

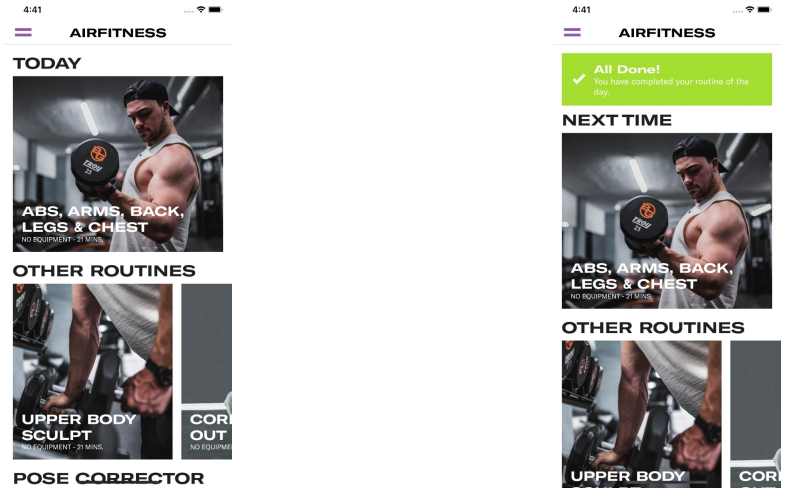
Figura 9: Vistas del proceso de registro: (a) Ingreso de información básica. (b) Selección del género. Aquellos que no se sientan identificados con el género masculino o femenino, lo pueden manifestar. (c) Elección de cuántas veces por semana entrenará



(a) **(b)** **(c)** **(d)**

Figura 10: Vistas del proceso de registro: (a) Nivel deportivo. (b) Objetivos. (c) Deporte. (d) Equipamiento.

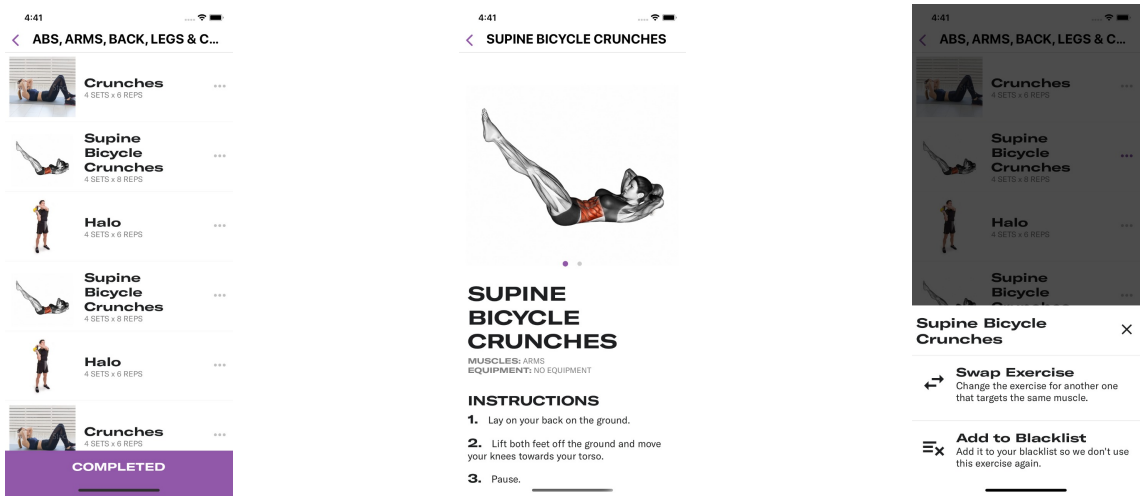
Pantalla principal



(a) **(b)**

Figura 11: Vista principal: (a) El usuario, al iniciar podrá ver sus rutinas, las rutinas predefinidas y acceder al corrector de ejercicios. (b) Al terminar la rutina asignada para el día, se le notifica que la ha completado.

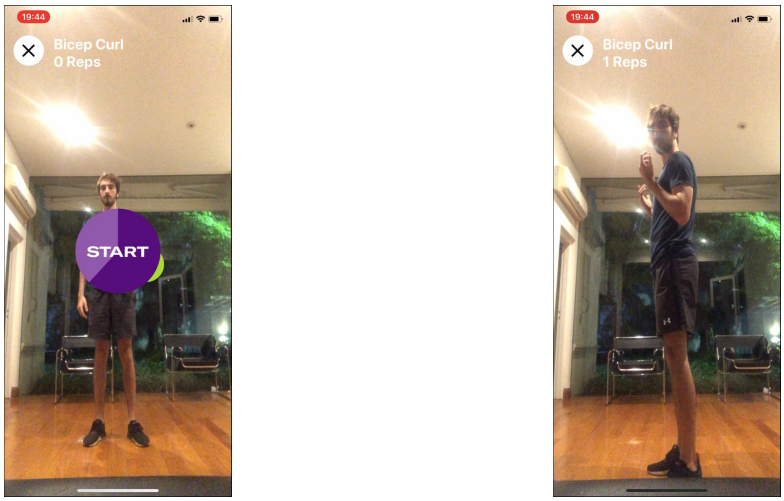
Rutinas



(a) **(b)** **(c)**

Figura 12: Vistas de las rutinas: (a) Detalle. (b) Instrucciones de un ejercicio. (c) Cambiar el orden o lista negra.

Corrección de ejercicios



(a) **(b)**

Figura 13: Vistas de corrección de ejercicios: (a) *START*. (b) Realiza el ejercicio.

6.2. Backend y Base de Datos

Para el *backend* contabamos con experiencia limitada ya que a lo largo de la carrera solo utilizamos Spring en Proyecto de Aplicaciones Web. Por lo tanto, realizamos una investigación sobre las nuevas tecnologías adoptadas por las empresas y decidimos desarrollar un servidor en Node.js con un API en GraphQL.

GraphQL es un nuevo estándar *open-source* de API que provee una alternativa más eficiente, poderosa y flexible en comparación a REST.

REST fue una forma popular de interactuar con el servidor. Cuando este estilo se desarrolló, los clientes eran simples. Sin embargo, actualmente la situación es muy diferente ya que la complejidad de los mismos y las expectativas de los usuarios de recibir nuevas funcionalidades y mejoras continuas se han incrementado.

GraphQL permite interactuar con el servidor de forma declarativa, especificando exactamente que se necesita mediante un API. En lugar de utilizar múltiples *endpoints* que devuelven estructuras de datos fijas, el servidor GraphQL expone un único *endpoint* y responde solo con lo que el cliente le pidió, minimizando la cantidad de información transferida. Fue desarrollada por Facebook y adoptada por miles de compañías en el mundo como Shopify, GitHub, Twitter, Pinterest, Yelp para nombrar algunas.

A la hora de desarrollar el *backend* es importante implementar una forma segura, bien organizada y performante para acceder a la base de datos. Una solución es utilizar un *data access layer* dedicado (DAL) que abstrae las complejidades del acceso a la base de datos. El servidor utiliza el API del DAL para comunicarse con la base de datos permitiendo poner el foco en la información que se quiere obtener en lugar de cómo obtenerla.

Al utilizar DAL, se consigue una clara separación de intereses en donde cada sección se enfoca en un interés determinado, eso permite mayor mantenibilidad, reusabilidad del código y asegura que el servidor pueda comunicarse con la base de datos de una forma segura y performante.

Para lograr esto, se decidió emplear Prisma. Prisma es un DAL auto-generado que utiliza los mismos principios que los DALs implementados en las empresas más importantes como Strato de Twitter o TAO de Facebook pero a su vez es accesible para aplicaciones más simples como AirFitness.

Prisma es compatible con PostgreSQL, MySQL, MongoDB, MariaDB, SQLite, AWS Aurora, Microsoft SQL Server y Azure SQL. Se optó por utilizar PostgreSQL ya que es la base de datos con la que el equipo posee mayor experiencia y considera que es adecuada para este tipo de aplicaciones.

Entrando en más detalle sobre el stack utilizado, el servidor está basado en Node.js, GraphQL, Yoga y como mencionamos anteriormente, Prisma.

En resumen, se diseñó la siguiente arquitectura:

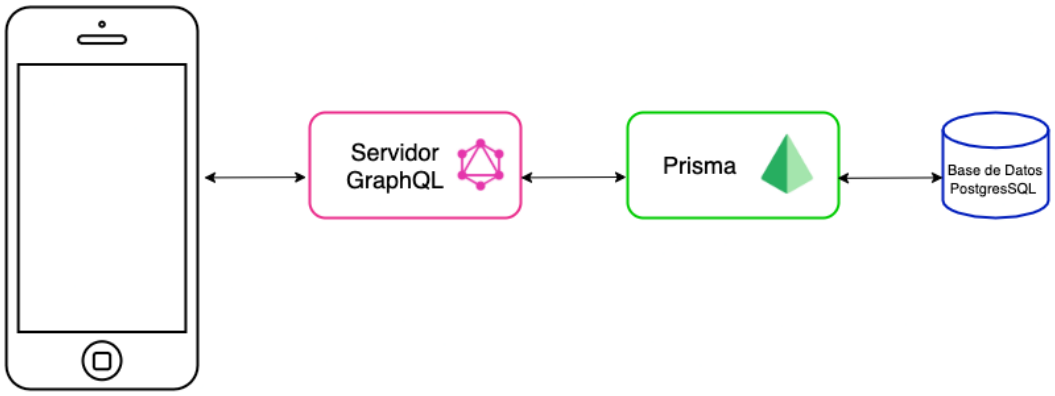


Figura 14: Arquitectura de alto nivel del sistema de software desarrollado.

En la Figura 14 comenzando desde la izquierda se encuentran:

- El cliente desarrollado en React Native.
- El servidor GraphQL que contiene toda la lógica de negocio, es decir todas las operaciones CRUD, autenticación, autorización, esquemas de base de datos, etc.
- Entre el servidor GraphQL y la base de datos, se encuentra Prisma. El mismo actúa como un DAL, lo que permite escribir queries SQL utilizando un paradigma orientado a objetos en algún lenguaje de nuestra preferencia, en este caso, Javascript.
- Por último, se encuentra la base de datos Postgres en donde se almacena la información.

7. Datos de implementación

En esta sección se darán a conocer detalles de las decisiones de implementación tomadas a la largo del desarrollo de la aplicación.

7.1. Corrección de ejercicios

Al encender la cámara PoseNet comienza a trabajar. Genera y aproxima coordenadas (x,y) de dieciséis puntos distintos a lo largo de todo el cuerpo, entre los que se pueden destacar: los codos, las muñecas, las rodillas, los tobillos, las caderas, entre otros.

Además, cada punto tiene un *score*. Dicho *score* refleja la confianza de la aproximación dada. En caso de que existan puntos que no son alcanzados por la cámara, el *score* es muy bajo, tiende a cero.

Con las coordenadas de cada articulación o parte del cuerpo se generaron vectores para facilitar el cálculo de los ángulos. Las fórmulas matemáticas utilizadas para llevar a cabo esta tarea fueron vistas en los primeros años de la carrera de Ingeniería Informática y se mencionan a continuación:

$$\begin{aligned} P_1 &= (x_1, y_1) \\ P_2 &= (x_2, y_2) \\ P_1P_2 = P_2 - P_1 &= (x_2 - x_1, y_2 - y_1) \end{aligned} \tag{1}$$

$$\theta = arccos\left(\frac{\vec{A}.\vec{B}}{|\vec{A}|.|\vec{B}|}\right) \tag{2}$$

Por ejemplo, para calcular el ángulo que se forma al flexionar el codo se tomaron tres puntos: muñeca, codo, y hombro. Se generaron dos vectores, uno que describe la parte superior del brazo (hombro/codo) y otra la parte inferior (codo/muñeca).

A partir de los vectores creados se puede obtener el ángulo con la fórmula (2) y en función de ello realizar el análisis correspondiente.

Existen ejercicios donde es obligatorio colocarse en forma paralela a la cámara. Por ejemplo, las sentadillas. Si el usuario fuese a realizar la sentadilla de frente a la cámara no se podría calcular el ángulo que se genera entre la pantorrilla y el muslo.

PoseNet no posee una forma de detectar de que lado se encuentra ubicado el usuario, por lo tanto para determinarlo se utilizó el *score* de las orejas, izquierda y derecha, tomando el mejor *score* para identificar como se encuentra ubicado el usuario.

Por último, la cámara un dispositivo móvil invierte la imagen por lo tanto si la oreja con mayor *score* es la derecha significa que el usuario está mostrando su lado izquierdo.

Dependiendo del ejercicio que se desea evaluar, se deben realizar y respetar una serie de restricciones en diferentes ángulos. Para detectar si el ejercicio está o no completo se separó a cada uno de ellos en tres etapas, una etapa inicial, que refiere a la postura inicial, un paso intermedio en donde los ángulos pueden variar entre un mínimo - máximo y una última etapa en donde se chequea que se cumplan todas las condiciones.

Si se cumplen las condiciones de la tercera etapa y luego se llega a una primera etapa, significa que el ejercicio ha finalizado y completó un ciclo. Existen ejercicios en donde esto se pudo lograr en dos etapas.

En este caso se llevó cabo el análisis de una serie de ejercicios que serán descriptos más adelante. El análisis de los ejercicios fue evaluado con profesionales ya que no existe una evaluación de los ángulos exactos que debe cumplir una persona para realizar un ejercicio correctamente.

7.1.1. Bicep Curl

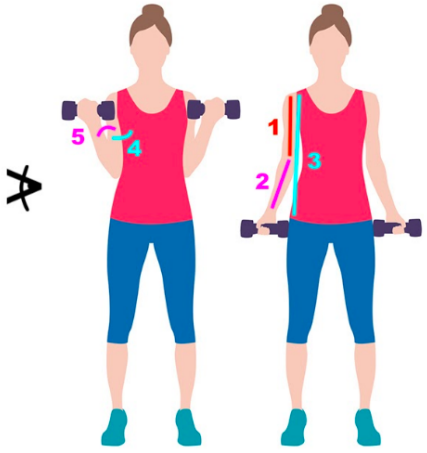


Figura 15: Bicep Curl

Se generaron tres vectores, dos de ellos describen el brazo: la parte superior (1) y la parte inferior (2) respectivamente. Estos, serán utilizados para conocer el ángulo que se forma al flexionar el brazo. El tercero describe el torso (3), la parte lateral que se extiende desde el hombro hasta la cadera. Este último, combinado con (1) permite definir si el usuario está despegando, o no, el brazo de su cuerpo.

Para determinar que el ejercicio esta siendo realizado correctamente se evaluarán dos ángulos. Por un lado, aquel que se forma entre el brazo y el cuerpo, es decir la parte superior del brazo (1) con el vector que describe el torso (3), en la Figura 15 indicado como el ángulo (4).

El brazo no debe despegarse del cuerpo en ningún momento, dado que lograr un ángulo de 0° es poco probable no solo porque no todos los cuerpos son iguales sino que además, la tecnología no es 100 % exacta. Se estableció que dicho ángulo no puede superar los 20°.

Por otro lado, el ángulo marcado en la Figura 15 como (5) describe aquel que se forma al doblar o plegar el brazo. Este es conformado por los vectores (1) y (2), la parte superior e inferior del brazo respectivamente. Para que el ejercicio se considere finalizado, dicho ángulo debe ser menor a 70°, al extender el brazo nuevamente y alcanzar un ángulo cercano a 180° damos por concluido el ejercicio.

En función del ángulo (5) se determinará si el ejercicio ha sido finalizado o no. Para determinar si el usuario cumplió con la etapa anterior y poder dar como terminado el ejercicio existe un indicador. Al llegar nuevamente a la etapa inicial y verificar dicho indicador se puede dar por concluido el ejercicio y aumentar una repetición.

Para que este ejercicio pueda ser fácilmente interpretado por la aplicación es necesario que el usuario se coloque de forma paralela a la cámara, es decir, en el caso de la Figura 15 la cámara debería estar a la izquierda del usuario. De esta forma, la aplicación podrá notar y verificar cada uno de los ángulos mencionados.

7.1.2. Squat

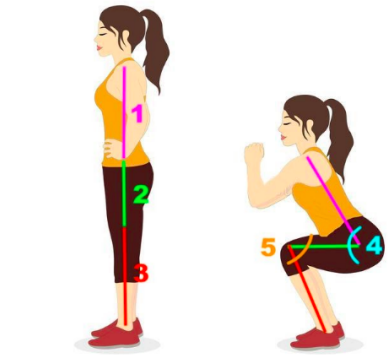


Figura 16: Squat o sentadilla

Nuevamente se generaron tres vectores, uno que describe el torso y al igual que el caso anterior se calcula tomando el punto del hombro y la cadera. Los últimos dos describen las piernas, muslo y pantorrilla. Cada uno de estos vectores está marcado en la Figura 16 con (1), (2) y (3) respectivamente.

En el caso de la sentadilla, se evaluaron dos ángulos: uno que hace referencia a la flexión de las rodillas (5) y otro, el ángulo que se genera entre las piernas y el torso (4). Para que el ejercicio esté realizado correctamente, al flexionarse, el ángulo de las rodillas debe variar entre 60° y 90°, no se permite que se exceda de los 60° porque podría estar poniendo en riesgo la salud de sus rodillas. Por otro lado, es importante saber si el usuario al realizar el ejercicio está inclinando su cuerpo hacia adelante perdiendo así la postura. Esto será controlado con el ángulo descripto en la Figura 16 como (4) y debe ser menor a 75° y mayor a 60°. Se separó el ejercicio en tres etapas:

1. **Paso 0**, etapa inicial:
   - *a*) $\theta_4 > 150°$
   - *b*) $\theta_5 > 160°$
2. **Paso 1**, comienza el ejercicio:
   - *a*) $\theta_4 < 150°$

3. **Paso 2**, se encuentra en la posición final:

   a) $60° < \theta_4 \leq 90°$

   b) $\theta_5 < 75°$

Como ha sido mencionado, debido a que la tecnología no es exacta y los cuerpos no son iguales, se ha realizado un exhaustivo análisis para conocer cuáles eran los ángulos correctos para completar el ejercicio y brindarle la mejor corrección al usuario.

7.1.3. Deadlift

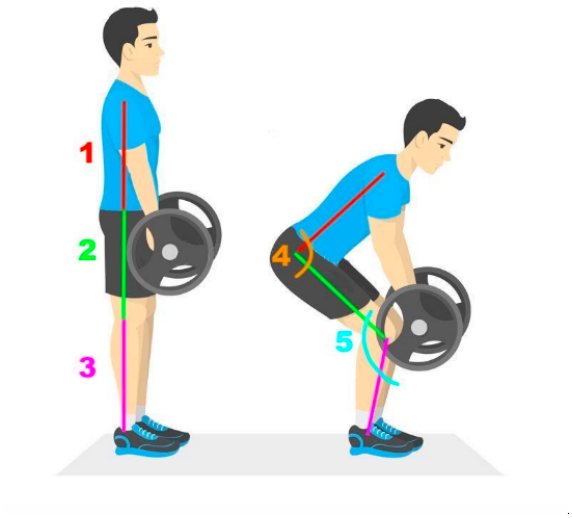


Figura 17: Deadlift o peso muerto

El peso muerto descripto en la Figura 17 puede ser fácilmente equivocado por una sentadilla. Si bien los vectores y ángulos a analizar son los mismos, el movimiento es totalmente diferente. Nuevamente, utilizamos un vector torso y dos vectores para describir las piernas. Dos ángulos, (4) y (5) que describen el ángulo torso-piernas y la flexión de piernas respectivamente.

Si bien la sentadilla y peso muerto tienen como principal objetivo fortalecer los cuádriceps, glúteos e isquiotibiales, su técnica es muy distinta. En la sentadilla, se busca llevar los glúteos hacia abajo como si se estuviese sentando. Por el contrario, al realizar peso muerto se busca generar la fuerza con el bicep femoral sin encorvar la espalda.

Al igual que otros ejercicios, este fue dividido en tres etapas:

1. **Paso 0**, etapa inicial:
   a) $160° \leq \theta_4 \leq 180°$
   b) $170° \leq \theta_5 \leq 180°$
2. **Paso 1**, comienza el ejercicio:
   a) $40° \leq \theta_4 \leq 180°$
   b) $110° \leq \theta_5 \leq 150°$
3. **Paso 2**, se encuentra en la posición final:
   a) $15° \leq \theta_4 < 40°$
   b) $110° \leq \theta_5 \leq 150°$. Este último no cambia ya que la fuerza se debe realizar con el bicep femoral.

7.1.4. Lateral leg kick

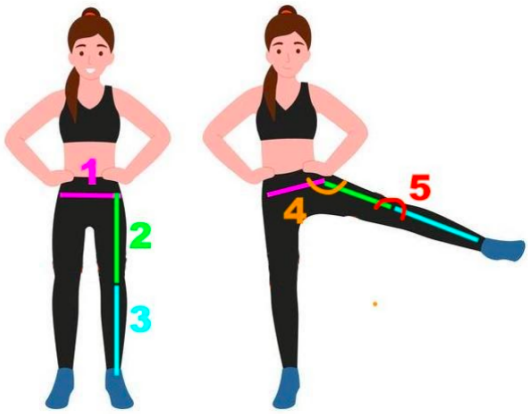


Figura 18: Lateral leg kick o patada lateral

Como se puede observar en la Figura 18 es un ejercicio que involucra los músculos de las piernas y el equilibrio. Para un correcto análisis del ejercicio se necesitan tres vectores. En primer lugar, el vector cadera que servirá para conocer la elevación de la pierna. En segundo lugar, un vector involucrando la parte superior de la pierna utilizada para el mismo fin y además, para conocer si la pierna está o no extendida. Por último, la parte inferior de la pierna que también será utilizada para analizar la extensión lograda.

A diferencia de los ejercicios anteriores de tren inferior, sentadilla y peso muerto en este caso, la cámara se debe colocar de forma frontal. Es decir, el usuario debe ubicarse frente a la cámara.

Las etapas en este caso son:

1. **Paso 0**, etapa inicial:
   - *a*) $0° \leq \theta_4 \leq 90°$
   - *b*) $160° \leq \theta_5 \leq 180°$
2. **Paso 1**, comienza el ejercicio:
   - *a*) $90° < \theta_4 \leq 180°$
   - *b*) $160° \leq \theta_5 < 180°$
3. **Paso 2**, se encuentra en la posición final:
   - *a*) $110° \leq \theta_4 \leq 180°$
   - *b*) $160° \leq \theta_5 \leq 180°$

Como se puede observar, en los 3 pasos se solicita que la pierna se mantenga siempre extendida, en caso de ser flexionada el ejercicio se marca como incorrecto y debe comenzar de cero.

7.1.5. Front Raise

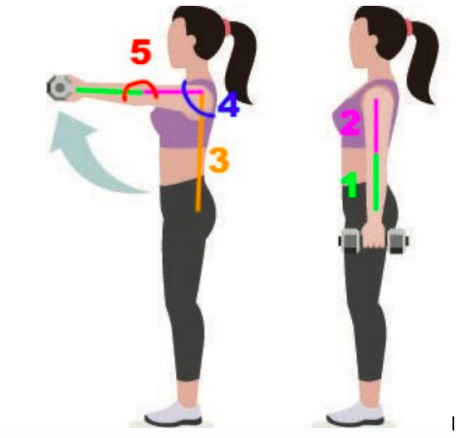


Figura 19: Front raise

Para el siguiente ejercicio, focalizado en hombros, se detectaron tres vectores principales: conformado por la parte inferior del brazo (1), el antebrazo que se extiende desde la muñeca hasta el codo, para la parte superior del brazo (2) conformado por los puntos indicados para el codo y el hombro; por último el torso (3) descripto por el hombro y cintura.

Al igual que el caso de la sentadilla, peso muerto y bicep curl, el usuario se debe parar de forma paralela a la cámara con el brazo que esta elevando del lado de la cámara. Esto es para poder comprobar la correcta postura del ejercicio.

Por otro lado, se identifican dos ángulos: la amplitud del brazo (4), el cual describe cuánto se eleva y su extensión (5). Para saber si el ejercicio se ha realizado de forma correcta hay que asegurarse que el brazo este siempre extendido y que al ser elevado no supere los 90°.

Como se ha mencionado, el ejercicio se ha dividido en tres etapas. El ejercicio es finalizado correctamente cuando transcurren las tres etapas y la aplicación nunca indicó un error en la postura.

En este caso las etapas fueron:

1. **Paso 0**, etapa inicial:
   a) $0° \leq \theta_4 \leq 5°$
   b) $160° \leq \theta_5 \leq 180°$
2. **Paso 1**, comienza el ejercicio:
   a) $5° < \theta_4 \leq 75°$
   b) $160° \leq \theta_5 \leq 180°$
3. **Paso 2**, se encuentra en la posición final:
   a) $85° \leq \theta_4 \leq 90°$
   b) $160° \leq \theta_5 \leq 180°$

El brazo no se debe flexionar nunca. Cuando el peso que levanta el individuo es excesivo no lo puede soportar y el brazo tiende a flexionarse pudiendo generarle una lesión.

7.1.6. Lateral Raise

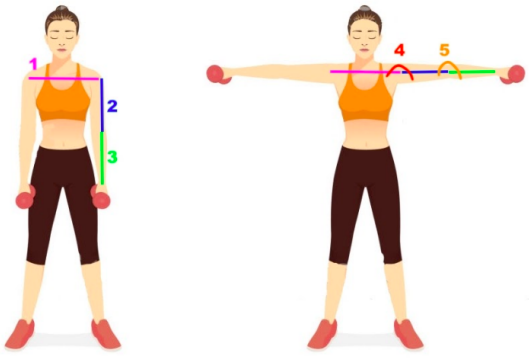


Figura 20: Lateral raise

Lateral raise es otro ejercicio de hombros. La idea de este es que ambos brazos sean elevados a la altura de los hombros, sin superarlos, buscando obtener una 'línea recta' de punta a punta. Para mayor simplicidad y lectura de la Figura 20, los vectores y ángulos involucrados fueron marcados en un solo brazo, pero para la resolución del ejercicio se tuvieron en cuenta ambos brazos en simultáneo. En total se evaluaron cinco vectores: cuatro de ellos involucran los brazos, parte superior e inferior de ambos (2)(3), izquierdo-derecho y el quinto conformado por los hombros (1).

El ejercicio involucra el análisis de cuatro ángulos, la elevación de ambos brazos, marcado en la Figura 20 como (4) y la extensión de estos como (5).

Al tratarse de un ejercicio que involucra los dos brazos es necesario que las condiciones que se marcarán a continuación se cumplan para ambos brazos, si uno de los brazos no lo respeta, el ejercicio se considera incorrecto.

En este caso las etapas fueron:

1. **Paso 0**, etapa inicial:
   a) $\theta_4 \leq 110°$
   b) $160° \leq \theta_5 \leq 180°$
2. **Paso 1**, comienza el ejercicio:
   a) $110° \leq \theta_4 \leq 174°$
   b) $160° \leq \theta_5 \leq 180°$
3. **Paso 2**, se encuentra en la posición final:
   a) $174° < \theta_4 \leq 180°$
   b) $160° \leq \theta_5 \leq 180°$

Para que el ejercicio sea correctamente detectado se le solicita al usuario colocarse de frente a la cámara.

7.1.7. Overhead Press

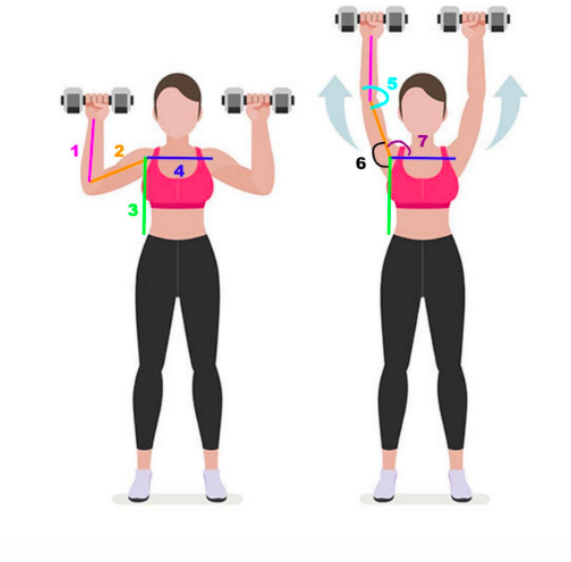


Figura 21: Overhead press

En este ejercicio se trabaja pectorales, triceps y hombros. Para completar y llevar a cabo un correcto análisis se requieren siete vectores: dos en cada brazo, uno para la parte superior (2) y otro para la parte inferior (1), un quinto vector que describe los hombros (4), y por último otros dos vectores para el torso generado por los puntos del hombro y la cadera (3), también de ambos lados.

Al igual que el caso anterior, para que el ejercicio sea correctamente ejecutado es necesario que ambos brazos cumplan con todas las reglas en simultáneo. Esto último es muy importante ya que, de no hacerlo, el ejercicio no solo no estaría completo sino que además, estaría realizado de forma incorrecta.

El ejercicio tiene una primera etapa en la que el usuario se encuentra parado con los brazos flexionados conformando un ángulo de aproximadamente 90°. Para completar el ejercicio debe extender ambos brazos por arriba de sus hombros, además, debe lograr una elevación total del brazo. Para un mayor entendimiento se describen las etapas y cuales son las condiciones que deben cumplir los ángulos indicados en la Figura 21 en cada caso.

1. **Paso 0**, etapa inicial:
   a) $80° \leq \theta_5 \leq 95°$
   b) $85° \leq \theta_6 \leq 95°$
   c) $175° \leq \theta_7 \leq 185°$
2. **Paso 1**, comienza el ejercicio:
   a) $95° < \theta_5 < 160°$
   b) $95° \leq \theta_6 < 150°$
   c) $115° \leq \theta_7 < 175°$
3. **Paso 2**, se encuentra en la posición final:
   a) $160° \leq \theta_5 < 185°$
   b) $150° \leq \theta_6 \leq 180°$
   c) $90° \leq \theta_7 < 115°$

Nuevamente la cámara debe colocarse de frente.

7.1.8. Triceps Extension

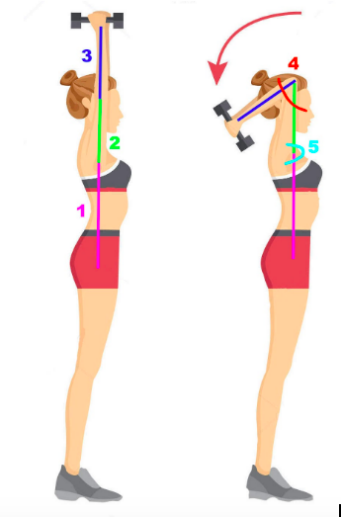


Figura 22: Triceps Extension

En este caso, los vectores mostrados en la Figura 22 son: el torso (1), y dos que involucran el brazo, uno describe su parte superior (2) y otro, su parte inferior (3). Los ángulos a evaluar son: el que se forma al flexionar el brazo (4) y un segundo ángulo para describir la extensión del mismo (5).

Para llevar a cabo el ejercicio de forma correcta se inicia con los brazos estirados, estos se deben llevar hacia atrás sin exceder determinado ángulo. Al finalizar el ejercicio, se debe volver a la etapa inicial, con los brazos extendidos.

La cámara se debe colocar de forma paralela al usuario para que se puedan distinguir cada uno de los vectores y ángulos ya mencionados.

Las etapas en este caso son las siguientes:

1. **Paso 0**, etapa inicial:
   - *a*) $170^\circ \leq \theta_4 \leq 180^\circ$
   - *b*) $160^\circ \leq \theta_5 \leq 180^\circ$
2. **Paso 1**, comienza el ejercicio:
   - *a*) $91^\circ < \theta_4 < 170^\circ$
   - *b*) $170^\circ \leq \theta_5 \leq 180^\circ$
3. **Paso 2**, se encuentra en la posición final:
   - *a*) $60^\circ \leq \theta_4 \leq 91^\circ$
   - *b*) $150^\circ \leq \theta_5 \leq 180^\circ$

En caso de estar flexionando excesivamente los brazos (4) o estar separando los brazos del cuerpo (5) la aplicación le indicará al usuario su error y cómo mejorarlo.

7.1.9. Scarecrow Extensions

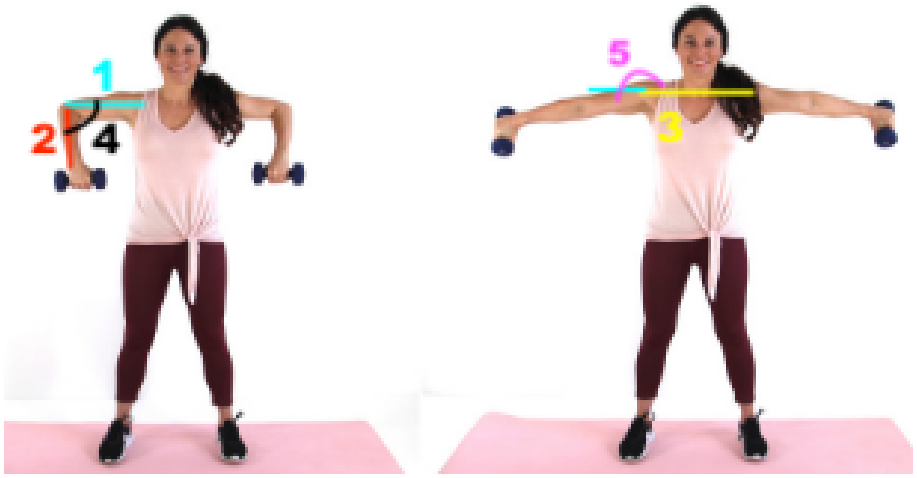


Figura 23: Scarecrow Extensions

Por último, se evaluará un ejercicio focalizado en la espalda. Su objetivo es que a partir de una posición inicial, en donde los brazos están formando un ángulo de 90° en sus codos y poder estirarlos sin afectar la postura. Este ejercicio requiere sincronismo en los brazos, es decir, ambos brazos deben hacer lo mismo y cumplir con las normas impuestas, en caso contrario, el ejercicio se está realizando de forma incorrecta.

En este ejercicio son necesarios cinco vectores: el vector que se genera entre ambos hombros (3), los brazos, izquierdo-derecho, con sus respectivos vectores (1)(2). Estos últimos sirven para poder calcular y conocer el movimiento de los brazos con mayor precisión ya que sino, por ejemplo, no se detectaría cuando se flexiona el brazo.

Dado que el foco está en poder flexionar y extender los brazos manteniendo una postura derecha los ángulos a evaluar son aquellos que se forman al flexionar los brazos, marcado en la Figura 23 como (4) y los que se generan entre la parte superior de los brazos con el vector compuesto por los hombros (5). Este último se busca mantener constante.

Para separar el ejercicio en etapas tuvimos las siguientes consideraciones:

1. **Paso 0**, etapa inicial:
   a) $85° \leq \theta_4 \leq 95°$
   b) $175° \leq \theta_5 \leq 185°$
2. **Paso 1**, comienza el ejercicio:
   a) $95° < \theta_4 \leq 175°$
   b) $175° \leq \theta_5 \leq 185°$
3. **Paso 2**, se encuentra en la posición final:
   a) $175° < \theta_4 \leq 185°$
   b) $175° \leq \theta_5 \leq 185°$

Es importante destacar que los ejercicios no fueron seleccionados al azar, si un usuario fuese a completarlos en una misma rutina tendría la mayor cantidad de músculos habitualmente entrenados.

7.2. Generación de Rutinas

Para la generación de rutinas se analizaron alternativas, Simulated Annealing y Greedy. Ambos son algoritmos de optimización pero su lógica a la hora de componer la solución es distinta, la cual podría brindar resultados diferentes a la hora de aplicarlos para resolver la problemática de la generación de rutinas.

Para conocer cual se adapta mejor al problema se pusieron a prueba. Se llevó a cabo una experiencia de dos semanas de duración en donde usuarios voluntarios probaron las rutinas generadas y expresaron su conformidad o disconformidad frente a un algoritmo en particular. Además, a partir de la información arrojada en dicho experimento se lograron definir nuevas reglas que sin dudas impactarían en la experiencia del usuario.

Antes de comenzar con la implementación de cada uno de los algoritmos se plantearon una serie de condiciones llamadas reglas. Si estas se cumplen, se ha logrado generar una rutina que no solo se adapta a las preferencias del usuario sino que, también cumple las condiciones planteadas por los profesionales.

Las conclusiones obtenidas a partir de esta experiencia, no solo nos permitieron modificar y/o mejorar las reglas que ya habíamos definido para el algoritmo, sino que también nos obligaron a contemplar nuevas reglas.

Para poder explicar cómo se implementó cada uno de los algoritmos, es importante describir qué información del usuario se tuvo en cuenta y fundamentalmente qué elementos componen una rutina.

Una rutina, es un conjunto de ejercicios que busca encontrar un balance entre las preferencias del usuario y las condiciones que se deben dar para que esté bien formulada. Por ejemplo, no deben existir dos ejercicios de alta dificultad consecutivos. Para completar esto último los ejercicios tienen asociadas una serie de propiedades.

Un ejercicio está compuesto por: un nombre, una lista de músculos para el cual dicho ejercicio puede ser utilizado, un nivel de dificultad, cuando corresponda el equipamiento necesario para poder realizarlo, un indicador de si el mismo requiere saltar o no y por último, instrucciones, imágenes y vídeo para una mejor comprensión.

Ejemplo:

```
{ id: 1, name: 'Squat', instructions: 'Squat', muscles: ['legs'],
image: 'Squat' , video:'Squat', equipment: [ ], difficulty: 1, jump:false }
```

Independientemente del algoritmo a utilizar, antes de comenzar la generación de rutina se define, de forma aleatoria, un número entre 8 y 10 inclusive el cual representa la cantidad de ejercicios que la rutina deberá contener. De este modo, se garantiza que las rutinas no sean excesivamente largas, logrando una duración de aproximadamente 45 a 60 minutos. Esta duración está vinculada con el nivel de dificultad establecida por el usuario, a mayor dificultad más repeticiones.

Recordemos que el usuario al inscribirse completa un conjunto de datos, entre ellos, sus preferencias, su objetivo, cuantos días va a entrenar y que equipamiento tiene.

En base a la cantidad de días a entrenar y su objetivo, se define para cada día un porcentaje del total de los ejercicios que deben pertenecer a cada músculo. Los músculos que pueden intervenir son: *abdominales, brazos, espalda, pecho, piernas, hombros, y cardiovascular/cardio.*

Para distribuir la carga de los músculos a entrenar no se tienen ejercicios de cada uno de ellos todos los días. Por ejemplo, si el usuario desea entrenar tres veces por semana y su objetivo es bajar de peso, se define un día para abdominales, brazos y cardiovascular, mientras que el segundo día, piernas y cardiovascular. El tercer día, abdominales y brazos.

Supongamos { `abdominales: 0.40, brazos: 0.20, espalda: 0.00, pecho: 0.00, piernas: 0.00, hombros: 0.00, cardiovascular: 0.40` }. El valor asociado a cada músculo expresa el porcentaje del total de ejercicios que se deben incluir en la rutina. En este caso, 40 % de los ejercicios deben ser abdominales, 20 % brazos y 40 % ejercicios cardiovascular.

En aquellos casos donde el porcentaje multiplicado por la cantidad total de ejercicios no resulte en un número entero, se decidió tomar el piso del número resultante.

Continuando con el ejemplo anterior, la rutina tendrá tres ejercicios de abdominales, uno de brazos y tres de cardiovascular. La suma de estos ejercicios da siete, por lo tanto restan reasignar dos ejercicios.

Para resolver dicho problema se implementó una nueva variable llamada *fav*. A esta variable se le asigna un musculo según el día y objetivo seleccionado. En caso de faltar ejercicios para cubrir el total, la falta será suplida por ejercicios que contengan dicho músculo.

En el ejemplo planteado, *fav* es `brazos` por lo tanto la rutina tendrá: tres ejercicios de abdominales, tres de brazos y tres cardiovasculares.

Inicialmente se filtran los ejercicios que no cumplan con las siguientes condiciones: el nivel deportivo del usuario y el equipamiento disponible. Los usuarios que hayan indicado que su nivel deportivo es principiante podrán realizar ejercicios de dificultad inicial/media. Si su nivel es intermedio o avanzado, ejercicios de dificultad media/avanzada.

Para la implementación del algoritmo basado en Simulated Annealing se utilizó una librería implementada en JavaScript: `'simulated-annealing'`. Esto permitió trabajar de forma individual sobre las reglas y su implementación sin tener que evaluar como implementar dicho algoritmo.

Para la generación del estado inicial se toman, de forma aleatoria, N ejercicios del listado previamente filtrado, siendo N un valor entre 8 y 10.

Según el nivel deportivo del usuario se define un máximo y un mínimo para la cantidad de repeticiones por ejercicio que podrá realizar:

1. **Principiante**
   a) min: 6
   b) máx: 10
2. **Intermedio**
   a) min: 14
   b) máx: 24
3. **Avanzado**
   a) min: 20
   b) máx: 30

En cada iteración **Simulated Annealing** generará una nueva solución. Dado que este algoritmo busca minimizar la función de costo/energía, la solución será evaluada en cada una de las reglas. Si cumple la regla suma 0 al total, caso contrario suma M. Dicho M depende de cuan importante o cuanto peso se le quiere dar a esa regla en particular. La función `getEnergy()` devolverá la suma total.

Inicialmente se generaron 8 reglas, todas reciben como parámetro la rutina entera, con sus N ejercicios. Las reglas serán detalladas a continuación.

- `checkExercise(routine)`
  - Itera sobre todos los ejercicios y chequea que no haya dos ejercicios iguales consecutivos. Dos ejercicios son iguales cuando su nombre es el mismo.
  - Si no cumple con la regla, devuelve 15.
- `checkRepetitions(routine)`
  - Al iterar sobre la rutina, en cada ejercicio chequea que la cantidad de repeticiones que se solicitan cumplan con el nivel deportivo del usuario. Es decir, si es un nivel inicial no puede exceder las diez repeticiones.

  - Por cada ejercicio que no cumple con la cantidad de repeticiones, suma uno y luego *devuelve* la suma. Por lo tanto, devuelve la cantidad de ejercicios que no cumplen con la cantidad de repeticiones indicada.

- `checkHardLevel(routine)`

  - Recorre los ejercicios de la rutina en busca de ejercicios cuya dificultad sea tres (avanzado) detectando si los mismos son consecutivos.
  - Por cada par de ejercicios difíciles consecutivos se suma cinco y luego devuelve dicha suma.

- `checkEvenRepetitions(routine)`

  - Considerando que existen ejercicios que pueden ser realizados de ambos lados del cuerpo, derecho e izquierdo, se incluyó una condición en que los ejercicios tengan números pares.
  - Por cada ejercicio cuyas cantidades de repeticiones no sean par suma cuatro y luego devuelve dicha suma.

- `checkBeginnerLevel(routine)`

  - Los usuarios que eligieron ser principiantes deben recibir ejercicios de nivel inicial e intermedio. Para poder mantener un balance entre ejercicios de ambos niveles se solicita que exista la mitad de cada uno.
  - Si no se cumple se devuelve cinco.

- `checkVariability(routine)`

  - Chequea que la cantidad de ejercicios del músculo sea igual a la cantidad de ejercicios planteados inicialmente para dicho músculo.
  - Si existe al menos un músculo que no lo cumple, devuelve cinco.

- `checkHardLevelBalance(routine)`

  - Esta función está enfocada para aquellos usuarios que establecen que su nivel deportivo es avanzado. Se chequea que exista aproximadamente la misma cantidad de ejercicios fáciles y difíciles.
  - Si no se cumple, devuelve veinte.

Los valores a devolver en cada función fueron obtenidos luego de varias instancias de pruebas.

Como ya se mencionó previamente también se evaluó el algoritmo **Greedy**. No se encontró ninguna librería que permitiera facilitar al implementación del mismo, así que se tuvo que implementar completamente. La diferencia con Simulated Annealing es que Greedy genera la solución de forma incremental, es decir, en vez de tomar un conjunto de ejercicios y chequear si cumplen con las reglas pautadas, agrega solo a aquellos ejercicios que cumplan con las condiciones. De esta forma, al finalizar su ejecución se puede afirmar que todos los ejercicios cumplen con lo pautado, algo que no podemos afirmar con Simulated Annealing.

Al igual que el caso anterior se define un mínimo y un máximo para la cantidad de repeticiones que admite cada nivel deportivo. Los valores definidos son los ya mencionados. Para la inicialización del algoritmo se agrega un ejercicio que cumpla con las condiciones básicas, que incluya un músculo de los involucrados en la rutina, que la cantidad de repeticiones respete el máximo/mínimo y que sea un valor par. En función de este primer ejercicio se construye la solución final.

Se comienza con el proceso de generar la rutina o solución final iterando tantas veces como sea necesario hasta conseguir que la solución tenga la cantidad de ejercicios que se pautaron inicialmente y que cumplan con todas las reglas definidas. Recordemos que dicho número puede variar entre 8 y 10 inclusive.

Los ejercicios pueden tener más de un músculo asociado. Estos músculos pueden variar desde abdominales hasta cardiovasculares. Cuando se selecciona un ejercicio en particular se debe definir con cual de todos los músculos para el cual puede ser utilizado va a ser elegido. Es decir, si el `ejercicio x` puede ser utilizado para `['abdominales', 'brazos', 'cardiovascular']` se va a elegir uno de ellos al azar. Por ejemplo, si se define usar el músculo `abdominales`, será considerado como tal y no será contabilizado como un ejercicio de brazos o cardiovascular.

Como ya se mencionó anteriormente, los ejercicios que se agregan a la solución son ejercicios que cumplen con todas las reglas pautadas. A continuación se darán a conocer las reglas desarrolladas.

Algunas reglas poseen el mismo propósito que las mencionadas en Simulated Annealing, por lo tanto no profundizaremos en ellas.

- `checkVariability(routine, muscles_exercise, muscle_id, exercise)`
  - **Parámetros** que recibe la función:
    - `routine` = solución alcanzada hasta el momento.
    - `muscles_exercise` = cantidad de ejercicios que admite la rutina para cada músculo.
    - `muscle_id` = músculo seleccionado dentro del conjunto de músculos que admite el ejercicio.
    - `exercise` = si el ejercicio que se está evaluando puede o no ingresar a la solución. Este último incluye, id del ejercicio, músculo, dificultad y cantidad de repeticiones.
  - **Objetivo**: Verificar que la cantidad de ejercicios por músculos está siendo respetada.
  - En el caso de no cumplir, devuelve `false`.
- `checkExercise(routine, exercise)`
  - **Parámetros** que recibe la función:
    - `routine` = solución alcanzada hasta el momento.
    - `exercise` = si el ejercicio que se está evaluando puede o no ingresar a la solución.
  - **Objetivo**: Chequear que no existan dos ejercicios iguales en la rutina.
  - En el caso de no cumplir, devuelve `false`.
- `checkHardLevel(routine, exercise)`
  - **Parámetros** que recibe la función:
    - `routine` = solución alcanzada hasta el momento.
    - `exercise` = si el ejercicio que se está evaluando puede o no ingresar a la solución.

- **Objetivo**: Chequear que no haya dos ejercicios de nivel avanzado consecutivos.
- En el caso de no cumplir, devuelve `false`.

- `checkEvenRepetitions(exercise)`
  - **Parámetros** que recibe la función:
    - `exercise` = ejercicio que se está evaluando si puede o no ingresar a la solución.
  - **Objetivo**: Chequear que los ejercicios tengan una cantidad par de repeticiones.
  - En caso de no cumplir, devuelve `false`.

- `checkRepetitions(exercise, level)`
  - **Parámetros** que recibe la función:
    - `exercise` = ejercicio que se está evaluando si puede o no ingresar a la solución.
    - `level` = nivel deportivo del usuario.
  - **Objetivo**: Verificar que la cantidad de repeticiones del ejercicio indicado coincide en el rango permitido por el mínimo y máximo de ese nivel.
  - En caso de no cumplir, devuelve `false`.

- `checkBeginnerLevel(routine, exercise, level)`
  - **Parámetros** que recibe la función:
    - `routine` = solución alcanzada hasta el momento.
    - `exercise` = ejercicio que se está evaluando si puede o no ingresar a la solución.
    - `level` = nivel deportivo del usuario.
  - **Objetivo**: Busca verificar que en caso de que el nivel usuario sea principiante exista una distribución equitativa entre ejercicios de dificultad fácil e intermedio.
  - En caso de no cumplir, devuelve `false`.

- `checkJumping(routine, exercise, cant)`
  - **Parámetros** que recibe la función:
    - `routine` = solución alcanzada hasta el momento.
    - `exercise` = ejercicio que se está evaluando si puede o no ingresar a la solución.
    - `cant` = cantidad total de ejercicios que puede incluir la rutina.
  - **Objetivo**: chequear que no existan más de 2 o 3 ejercicios que involucren saltar. Si la cantidad de ejercicios de la rutina es 10, se permiten hasta 3 caso contrario 2. Esto surgió luego de la experiencia, ya que algunos usuarios de niveles avanzados debían realizar más de 5 o 6 ejercicios que involucraban saltos y sus repeticiones eran muchas, lo que potencialmente podría perjudicar sus espaldas.
  - En caso de no cumplir, devuelve `false`.

Al finalizar, se obtiene una rutina que cumple con todas las reglas pautadas.

Anteriormente hemos mencionado la realización de un experimento el cual permitió comprender y establecer cuáles eran las reglas que se debían implementar, cuales mejorar y fundamentalmente con qué algoritmo de optimización se debía trabajar.

Se contactó a 20 personas. A cada una de ellas se les envió un cuestionario compuestas por las mismas preguntas que se les hacen a los usuarios al inscribirse: información básica, nivel deportivo auto percibido, cantidad de veces a entrenar por semana, objetivo y equipamiento deportivo.

El experimento tuvo una duración de dos semanas. Durante la primera semana, los usuarios utilizaron las rutinas generadas por Simulated Annealing y la siguiente, las rutinas generadas por Greedy. Ninguno sabía que existían dos algoritmos y mucho menos que estos se comportan de forma diferente. Además de determinar cuál es el mejor algoritmo, el experimiento sirvió para conocer que opinaban los usuarios sobre la dinámica, tiempos de descanso, tipos de ejercicio, etc.

A partir de la información proporcionada por los usuarios, se les generaron rutinas adaptadas a sus preferencias y necesidades. Fueron enviadas por email a cada uno. Al finalizar la rutina, el usuario debía completar una encuesta para darnos *feedback*.

Dicha encuesta contemplaba una serie de preguntas que permitieron comprender como fue la experiencia del usuario al realizar las mismas. Esta contemplaba preguntas como si pudieron o no finalizar la rutina, si consideraban que faltaban ejercicios para completar su objetivo, cuán difícil le resultó realizarla, si los períodos de descanso eran adecuados, entre otras.

A partir del *feedback* obtenido se pudo comprobar que en el nivel inicial de entrenamiento los tiempos de descanso entre series eran adecuados, pero no así con el nivel intermedio (30 % de los participantes intermedios) o avanzado (80 % de los participantes avanzados). Esto nos permitió concluir que era necesario aumentar dichos tiempos.

A pesar de haber incrementado los tiempos de descanso, se logró que la duración de la rutina no superara los 45 a 60 minutos de acuerdo al nivel de entrenamiento.

Algunos usuarios que realizaron rutinas de nivel avanzado mencionaron la necesidad de reducir las repeticiones, en algunos casos se alcanzaban las 40 repeticiones por ejercicio. También surgió la importancia de limitar la cantidad de ejercicios que involucren saltos dado que podría afectar sus rodillas o espalda.

Si bien los usuarios no demostraron una preferencia de un algoritmo por sobre el otro, lo que pudimos comprobar es que en ciertas ocasiones Simulated Annealing repetía los ejercicios o no respetaba algunas de la reglas planteadas.

En función de lo mencionado y que Greedy había sido 100 % implementado por el equipo, se decidió trabajar con este último.

En el *feedback* algunos usuarios comentaron que no podían realizar determinados ejercicios ya que tienen algún tipo de lesión o simplemente, no les gustaban. En función de esto se generó una nueva funcionalidad: el usuario podrá colocar en una lista negra aquellos ejercicios que no desean o no pueden realizar. Al colocarlos en dicha lista, los mismos ya no serán considerados durante la generación de futuras rutinas.

Realizar el experimento brindó el sustento empírico necesario para poder generar nuevas reglas, modificar las existentes y fundamentalmente conocer e interpretar a los usuarios con el fin de poder brindarles la mejor experiencia posible.

8. Prototipo

Al comenzar el proyecto se evaluó la posibilidad de que todos los usuarios utilicen, de forma obligatoria, el corrector de pose en todos los ejercicios que hacían con las rutinas generadas por la aplicación.

Sin embargo, después de analizarlo se concluyó que forzarle esto al usuario podría resultar en una experiencia negativa por varios motivos.

En primer lugar, es probable que los usuarios con experiencia ya sepan realizar los ejercicios y no necesiten que la aplicación se los corrija.

En segundo lugar, los usuarios sin experiencia, luego de realizar el ejercicio reiteradas veces van a haber aprendido a realizarlo correctamente. Además, existe la posibilidad de que el usuario no pueda utilizar la cámara por un impedimento del espacio físico en el que se encuentra.

Por estos motivos, se decidió hacer que la funcionalidad de la corrección de los ejercicios sea un módulo separado del de realizar las rutinas y solo aquellos usuarios que quieran usarlo para corregir un ejercicio tengan la posibilidad de hacerlo.

También se consideró la posibilidad de mostrar en la pantalla del dispositivo móvil las partes del cuerpo que no estaban realizando correctamente el ejercicio y/o como este debía realizarse. Por ejemplo, al hacer sentadillas la aplicación podría detectar que la espalda no está derecha y marcarle esa parte del cuerpo en rojo. Sin embargo, a la hora de hacer una prueba, se detectó que dicha funcionalidad iba a ser poco útil debido a la distancia que existe entre el usuario y el dispositivo móvil y la dificultad de tener que mirarlo mientras lleva a cabo el ejercicio.

Por lo tanto se decidió que sería más adecuado que el dispositivo móvil le indique por medio de la voz lo que no está haciendo de forma correcta. De esta manera en vez de marcar la espalda en rojo, la aplicación alertará al usuario diciendo “asegúrate de mantener la espalda recta”.

Al mismo tiempo, el contador de repeticiones puede ser difícil de leer, por lo tanto se implementó la misma solución que en el caso anterior. Al realizar una repetición de un ejercicio, el usuario va escuchar el número de repeticiones en la que se encuentra.

9. Evaluación

9.1. Especificación de casos de prueba

Tabla 1: Caso de prueba para la creación de un nuevo usuario

Evaluación	Crear una cuenta
Objetivo	Verificar que un usuario pueda registrarse en la plataforma
Entradas	1. Nombre 2. Apellido 3. Email 4. Contraseña 5. Contraseña repetida 6. Edad 7. Sexo 8. Altura 9. Peso 10. Objetivos 11. Deportes 12. Frecuencia 13. Nivel
Pasos	1. Ingresar a la aplicación 2. Hacer click en "Sign up" 3. Completar todos los pasos del formulario 4. Hacer click en "Sign up"
Resultado	**Exitoso**

Tabla 2: Caso de prueba para iniciar sesión con una cuenta existente

Evaluación	Iniciar Sesión
Objetivo	Verificar que un usuario pueda ingresar a su cuenta existente
Entradas	1. Nombre 2. Contraseña
Pasos	1. Ingresar a la aplicación 2. Hacer click en "Log in" 3. Completar los datos 4. Hacer click en "Log in"
Resultado	**Exitoso**

Tabla 3: Caso de prueba para agregar un ejercicio a la base de datos

Evaluación	Agregar ejercicio a la DB
Objetivo	Verificar que se pueda agregar un ejercicio a la base de datos
Entradas	1. Nombre 2. Instrucciones 3. Músculos 4. Imagen 5. Equipamiento 6. Dificultad
Pasos	Correr la función en el *backend*
Resultado	**Exitoso**

Tabla 4: Caso de prueba para generar una rutina predefinida

Evaluación	Generar rutina predefinida
Objetivo	Verificar que se pueda agregar una rutina predefinida a la base de datos
Entradas	Ejercicios
Pasos	Correr función en el *backend*
Resultado	**Exitoso**

Tabla 5: Caso de prueba para generar una rutina personalizada

Evaluación	Generar rutina personalizada
Objetivo	Verificar que se pueda generar una rutina personalizada para un usuario determinado
Entradas	ID usuario
Pasos	Correr función en el *backend*
Resultado	**Exitoso**

Tabla 6: Caso de prueba para identificar si la rutina esta completa

Evaluación	Completar rutina
Objetivo	Verificar que se pueda completa una rutina y que e le sume 1 punto al usuario que la completo
Entradas	1. ID usuario 2. ID rutina
Pasos	1. Ingresar a la aplicación 2. Ingresar al detalle de una rutina 3. Hacer click en completar
Resultado	**Exitoso**

Tabla 7: Caso de prueba para identificar el cambio de equipamiento de un usuario

Evaluación	Cambiar equipamiento
Objetivo	Verificar que se pueda cambiar el equipamiento y que se generen nuevas rutinas personalizadas con el nuevo equipamiento
Entradas	1. ID usuario
Pasos	1. Ingresar a la aplicación 2. Hacer click en el menu 3. Hacer click en *Change Equipment* 4. Seleccionar el nuevo equipamiento y guardar
Resultado	**Exitoso**

10. Conclusiones

Si bien se han logrado completar los objetivos se considera que las tecnologías utilizadas podrían ser mejoradas o inclusive sustituidas por otras que logren detectar con mayor precisión otros movimientos, por ejemplo, si el usuario estuviese al nivel del piso.

Cuando se portó la implementación realizada con la cámara de la computadora a la del dispositivo móvil, se detectaron algunos problemas de detección en los movimientos. Esto devino en tener que realizar un *issue en GitHub*. Afortunadamente, su respuesta, pudo resolver el problema y de esta forma se consiguió identificar los mismos ejercicios ya evaluados con la cámara de la computadora en el dispositivo móvil.

Si bien se logró implementar un *'personal trainer'* virtual, la idea es que la aplicación no suplante a los profesionales sino que sea un complemento de tarea que estos realizan. Si bien los objetivos inicialmente planteados son bajar de peso, tonificar o mejorar la masa muscular, en un futuro estos podrían vincularse con la práctica de un deporte permitiendo utilizar la aplicación para analizar la biomecánica del cuerpo y conocer cómo los usuarios se comportan según el deporte que practican.

Por último, con una única iteración en el desarrollo del algoritmo generador de rutinas, se lograron detectar múltiples aspectos que pueden ser mejorados por lo tanto estamos seguros que de continuar con la evolución del mismos las mejoras que se podrían implementar son innumerables.

11. Trabajo Futuro

Consideramos que a pesar de todo el trabajo realizado todavía existe lugar para mejoras y nuevas funcionalidades. A continuación se detallan algunas de las que fueron evaluadas:

- Mejorar las reglas del algoritmo de generación de rutinas para sea aún más preciso. Por ejemplo, se podría agregar una regla que evite que se repitan ejercicios entre rutinas, es decir si el Lunes hizo sentadillas evitar que el Miércoles las vuelva a hacer.
- Poder aprovechar la cámara para para hacer ejercicios físicos mediante juegos interactivos. Por ejemplo, se podría hacer que aparezcan diferentes elementos en la pantalla y que el usuario tenga que golpearlos con las manos o pies. Considerando que la aplicación logra contar repeticiones, se podrían hacer desafíos, por ejemplo, repetir un ejercicio N veces.
- El usuario podría completar dentro de su registro si tuvo o no alguna lesión que le impida realizar determinados ejercicios. De esta forma se evita que los usuarios deban incluirlos de forma manual en una lista negra.
- Asociarlo a relojes inteligentes para poder tener un resumen de la rutina, cuántas calorías se quemaron, cuánto tiempo se ejercitó, etc.
- Con la finalidad de motivar a los usuarios se podría agregar una tabla de posiciones tal como tienen las aplicaciones estudiadas en el estado del arte. Fomentar, a través de la aplicación, una comunidad basada en el deporte que busca mejorarse y encontrar su mejor versión continuamente.

Referencias

[Bertsimas and Tsitsiklis, 1993] Bertsimas, D. and Tsitsiklis, J. (1993). Simulated annealing. *Statistical Science*, 8(1):10–15.

[Facebook, 2020] Facebook (2020). Graphql. `https://graphql.org/`.

[Fister and Fister, 2015] Fister, I. F. J. S. R. K. L. F. D. and Fister, I. (2015). Planning fitness training sessions using the bat algorithm. pages 1–6.

[FitCam, 2019] FitCam (2019). Fitcam: Fitness trainer. `https://apps.apple.com/ar/app/fitcam-fitness-trainer/id14366504613`.

[FitMe, 2019] FitMe (2019). Fitme - ai fitness coach. `https://apps.apple.com/us/app/fitme-ai-fitness-coach/id1464966903`.

[FitnessAI, 2018] FitnessAI (2018). Fitnessai. `https://www.fitnessai.com/`.

[Jaime Guzmán-Luna, 2015] Jaime Guzmán-Luna, Ingrid-Durley Torres, S. V. (2015). Un sistema recomendador móvil de rutinas de ejercicio basado en el perfil del usuario. *Universidad Nacional de Colombia, Medellín, Colombia*, pages 1–14.

[Mirror, 2020] Mirror (2020). Mirror. `https://www.mirror.co/`.

[OpenPose, 2018] OpenPose (2018). Openpose: Realtime multi-person 2d pose estimation using part affinity fields. `shorturl.at/hmozT`.

[Peloton, 2012] Peloton (2012). Peloton. `https://www.onepeloton.com/`.

[PoseNet, 2020] PoseNet (2020). Estimación de pose. `https://www.tensorflow.org/lite/models/pose_estimation/overview`.

[Prisma, 2020] Prisma (2020). Prisma. `https://www.prisma.io/`.

[Sharma, 2020] Sharma, A. (2020). Basics of greedy algorithms. `https://www.hackerearth.com/practice/algorithms/greedy/basics-of-greedy-algorithms/tutorial/`.

[suprnrdy, 2020] suprnrdy (2020). Exploration: Pose estimation with openpose and posenet. `shorturl.at/iqCX5`.

[Tempo, 2018] Tempo (2018). Tempo. `https://tempo.fit/`.

[VAY, 2020] VAY (2020). Human motion analysis vay switzerland. `https://www.vay.ai/`.

[Yang, 2018] Yang, S. C. R. (2018). Pose trainer: Correcting exercise posture using pose estimation. *Department of Computer Science, Stanford University*, pages 1–8.

[Yang, 2010] Yang, X.-S. (2010). Bat algorithm: Literature review and applications. *Int. J. Bio-Inspired Computation*, 5(3).

12. Anexos

- Encuestas de Experimento
  - Formulario de inscripción
  - Respuestas de inscripción
  - Formulario para feedback
  - Resultados de feedback