UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería Informática

Monitorización y análisis del estado afectivo de sujetos humanos mediante dispositivo móvil

Trabajo Fin de Grado

Grado en Ingeniería Informática

AUTOR/A: Raya López, Aarón

Tutor/a: Vivancos Rubio, Emilio Pedro

Cotutor/a: Taverner Aparicio, Joaquín José

Cotutor/a externo: BOTTI NAVARRO, VICENTE JUAN

CURSO ACADÉMICO: 2022/2023

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

etsinf

Escola Tècnica Superior d'Enginyeria Informàtica

Escola Tècnica Superior d'Enginyeria Informàtica
Universitat Politècnica de València

Monitorización y análisis del estado afectivo de sujetos humanos mediante dispositivo móvil

TRABAJO FIN DE GRADO

Grado en Ingeniería Informática

Autor: Aarón Raya López

Tutor: Emilio Pedro Vivancos Rubio
Joaquín José Taverner Aparicio

Curso 2022-2023

Resum

L'augment de l'esperança de vida ha comportat que un gran nombre de persones grans presenten signes de deteriorament en la salut emocional i cognitiva. En aquest deteriorament hi han influït múltiples factors socials i demogràfics. De tots ells, l'aïllament i la soledat involuntària semblen els més rellevants. La falta de contacte amb persones properes repercuteix directament en l'aparició de diverses malalties com l'ansietat i la depressió.
En aquest treball acadèmic es busca crear una eina que permetra al pacient reflectir el seu estat emocional a mode de diari, de manera fàcil, ràpida i poc intrusiva, de forma que els familiars i facultatius puguen tenir un historial del pacient que permeti detectar patrons entre activitats i emocions.

Paraules clau: Computació afectiva, salut mental, detecció d'emocions, reconeixement facial

Resumen

El aumento de la esperanza de vida ha traido consigo que un grán número de personas mayores presenten signos de deterioro en su salud emocional y cognitiva. En este deterioro han influido múltiples factores sociales y demográficos. De todos ellos, el aislamiento y soledad involuntaria parecen los más relevantes. La falta de contacto con personas cercanas repercute directamente en la aparición de diversas enfermedades como ansiedad y depresión.
En este trabajo académico se busca crear una herramienta que permita al paciente reflejar su estado emocional a modo de diario, de manera fácil, rápida y poco intrusiva, para que familiares y facultativos puedan tener un historial del paciente que permita detectar patrones entre actividades y emociones.

Palabras clave: Computación afectiva, salud mental, detección de emociones, reconocimiento facial

Abstract

The increase in life expectancy has meant that a large number of Older people present signs of deterioration in their emotional and cognitive health. goes. Multiple social and demographic factors have influenced this deterioration. Of In all of them, isolation and involuntary loneliness seem the most relevant. The lack of contact with close people directly affects the appearance of various illnesses such as anxiety and depression. This academic work seeks to create a tool that allows the patient reflect your emotional state as a diary, easily, quickly and non-intrussive, so that relatives and doctors can have a history of the patient that allows detecting patterns between activities and emotions

Key words: Affective computing, mental health, emotion detection, facial identification

Índice general

Índice de figuras

Índice de tablas

CAPÍTULO 1
Introducción

1.1 Motivación

La edad media de la población está en aumento, lo que requerirá centrar más los esfuerzos sanitarios en las personas mayores.

Durante las últimas décadas, las sociedades occidentales llevan experimentando un fenómeno sin precedentes denominado "regresión de la pirámide poblacional". Dicho fenómeno implica que, dada una baja natalidad prolongada en el tiempo, junto a un aumento en la esperanza de vida gracias tanto a los avances sociales, en materia de medicina y una tasa de mortalidad reducida, la población de un país tiende a congregarse en el rango etario más elevado.
La forma más fácil de interpretar esto es con el uso de las **pirámides poblacionales** (1.1). Dichas gráficas muestran en el eje vertical los diferentes rangos de edad, y el eje horizontal la proporción de cada uno de estos sobre la población total (los valores se toman absolutos. No hay que tener en cuenta el signo).

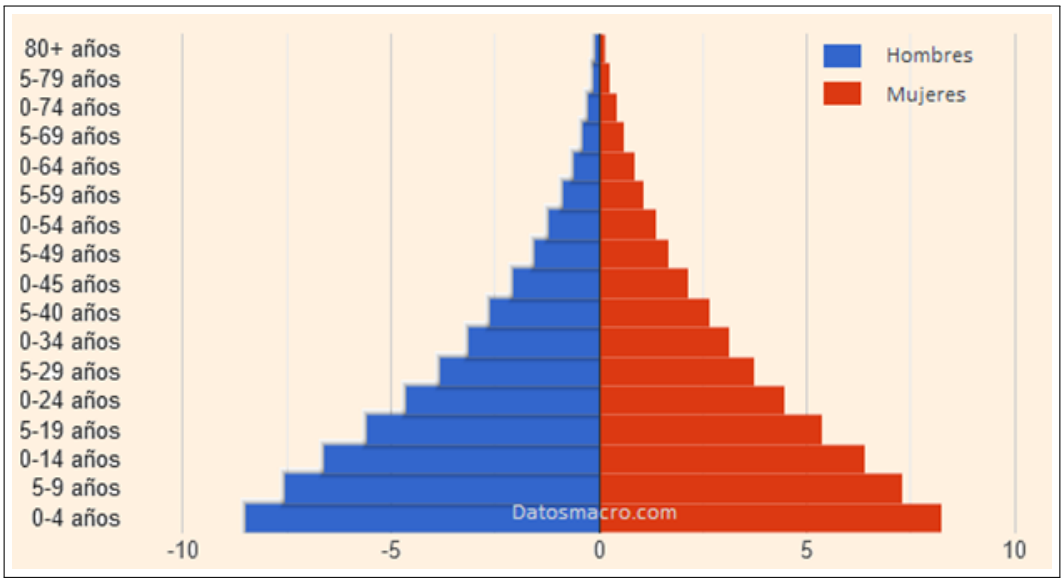

Figura 1.1: Pirámide poblacional teórica de ejemplo. Fuente [1]

¿Qué relevancia tiene esto? Al momento de redactar esta memoria, nos encontramos con que el número mayoritario de ciudadanos de nuestro país se encuentra en el rango de edad superior a los 50 años (véase el grupo mayoritario en el año 2022 en la figura 1.2, así como su desplazamiento para la predicción del año 2072). Esto significa que los sistemas

sanitarios van a tener que absorber un mayor volumen de pacientes en edades avanzadas de lo que se venía viendo hasta ahora. Y no se espera que esta tendencia vaya a cambiar a corto plazo.
En el artículo publicado por parte de los docentes Doris Cardona Arango y Enrique Peláez (Universidades CES Colombia y Córdoba Argentina, respectivamente) [13] se detalla un extenso análisis sobre las consecuencias y oportunidades que este fenómeno único puede suponer. En especial, haremos mención a una reflexión muy relevante con respecto al área del presente trabajo:

«Paradójicamente, una de las metas de los sistemas de salud, y específicamente de las instituciones de salud, es hacer que las demandas y las necesidades de los usuarios coincidan, pero son muy pocas las acciones realizadas para eliminar las barreras de accesibilidad que ponen los sistemas de salud a las personas, y en especial a aquellas menos favorecidas, y donde, desafortunadamente, los problemas de salud son más graves. Pero la gran brecha que hay entre demanda de servicios y necesidades de la población también indica la existencia de problemas culturales o desconocimiento de la problemática epidemiológica»[13]

El artículo fue publicado en el año 2012, pero la problemática no parece haberse diluido en el tiempo. No obstante, en los últimos años hay una corriente que enfatiza la necesidad de reforzar el tratamiento y cuidados a la tercera edad: creación de voluntariados para dar compañía a las personas en soledad, movilizaciones para la mejora e inversión de los sistemas de atención primaria, el apoyo a mayores de 65 años en comercios y administración pública, o los programas especiales a causa de la pandemia de SARS-CoV2) [14].
Se requiere reforzar los sistemas sanitarios con mayores medios y herramientas que faciliten a los profesionales el tratamiento de los pacientes pero si los gobiernos no optan por la opción de aumentar la inversión en los sistemas de salud, solo queda la opción de aumentar el rendimiento obtenido de la actual. Para ello se puede intentar conseguir:

1. **Mayor número de pacientes atendidos por facultativo**: Si delegamos ciertas tareas a herramientas automatizadas, con el mismo número de activos sanitarios, podemos ofrecer más dedicación a los pacientes, haciendo más útiles las horas invertidas por paciente, y requiriendo menos horas para cada uno, aumentando el volumen de pacientes tratados.

2. **Reorganización de costes**: Si no se aumenta el presupuesto dedicado a sanidad y atención a la tercera edad, no hay más remedio que conseguir un mejor uso de los recursos reorganizando la asignación a distintas partidas.

3. **Mejora en la calidad de la atención**: Las herramientas software pueden ofrecer un servicio constante para el usuario, además de enfocarse mejor en las necesidades de este. Es lo que se conoce como medicina personalizada, y se espera que tome un enfoque hacia esta dirección, lo que permitiría tratamientos con mejor respuesta en los pacientes, gracias a poder ajustarse a sus necesidades exclusivas.

Dentro de todas las enfermedades que se pueden experimentar en la vejez, nos centraremos en aquellas relacionadas con el campo de la **salud mental**. Las enfermedades mentales abarcan un amplio abanico de tipos, grados, afecciones y reacciones en cada individuo de forma diferente, como cabría esperar de un campo tan amplio como la psicología, psiquiatría y neurología. Estas afecciones pueden generar un gran impedimento para el desarrollo y disfrute de la vida. Incluso en casos severos, puede ser arrebatada, como es el caso de los suicidios. Centrándose en los casos más comunes, encontraremos el trastorno

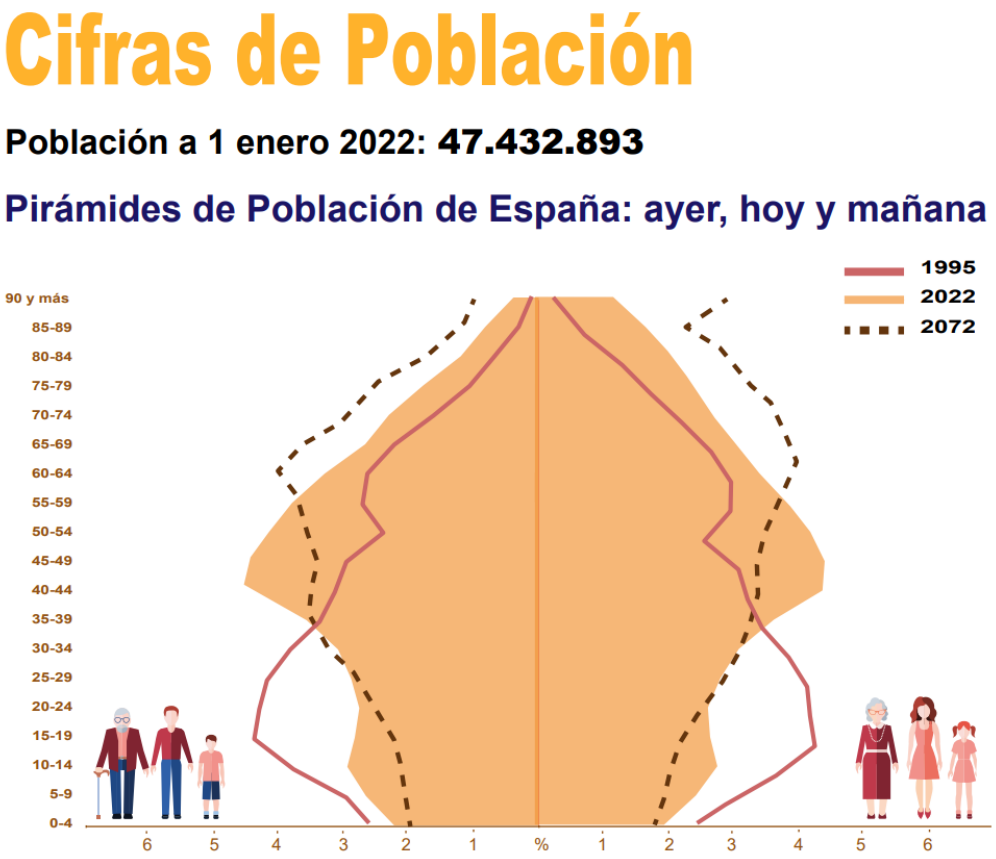

Figura 1.2: Pirámide poblacional de España en 2022. Fuente [2]

de ansiedad generalizada (TAG) y la depresión (en todas sus clasificaciones) como los más recurrentes diagnósticos médicos; ambas enfermedades, si bien son completamente diferentes, reúnen características que podemos observar a continuación:

1. **Pensamientos intrusivos**: Aparecen en el paciente de forma recurrente e involuntaria. Por lo general, son pensamientos negativos sobre la propia persona y el entorno, y dificultan la toma de decisiones de forma racional y objetiva
2. **Sintomatología física**: Como fatiga crónica, dolores estomacales, mareos etc.
3. **Aislamiento social**: Es habitual dejar de hacer las actividades que a uno le hacían feliz cuando uno está triste (depresión) o cuando dichas actividades ahora le generan malestar (ansiedad).

Junto a esto, aparece el distanciamiento entre el paciente y sus allegados, lo cual perjudica aún más el estado en el que ya se estaba. Además, mucha gente no tiene el conocimiento correcto de como tratar estas situaciones, lo que puede llevar al fin de la relación de forma involuntaria por parte del paciente por no poder gestionar sus emociones. ¿Qué puede provocar estas enfermedades? Existen muchos factores y contextos que pueden dar a pie la aparición de estas. Además, las enfermedades mentales tienen la característica de aparecer de una forma muy particular y específica para cada persona. Hay situaciones que una persona puede encontrar superior a sus capacidades, darles una gran importancia dentro de sus vidas o no verse capaz de afrontarlas, pero que para otra no suponen un problema más grave que cualquier otro corriente. Esto difiere mucho según el carácter de la persona, el ambiente en el que creció (círculo familiar, amistades, relaciones, etc.) e incluso su predisposición genética a adquirir cierta personalidad.

Aún así, podemos aventurarnos a recopilar una serie de elementos comunes entre un gran número de población que pueden explicar el crecimiento del número de pacientes. El principal de todos ellos es la **soledad involuntaria**. Definimos este término como la pérdida de contacto humano debido a factores en los cuales el paciente es forzado a aislarse de la sociedad, sin tener mucha capacidad de cambiarlo, o no encontrar los medios necesarios para evitarlo. Es importante recalcar que, si bien la soledad es un estado completamente normal en ciertos momentos de la vida, y que no tiene porqué implicar un perjuicio para la persona, nosotros haremos referencia a situaciones sin contacto social prolongado, que mermen la calidad de vida de la persona y que la situación en la que se encuentra no sea la deseada. A pesar de que muchas personas se mentalicen de que quieren estar solar, muestren prefieran o lo acepten como la única forma de vida que les quede, es necesario para un correcto desarrollo de las personas el mantener un mínimo de contacto con otros seres.

A modo de conclusión inicial, tenemos dos factores que motivan la realización del trabajo presentado en esta memoria: **población envejecida** y **soledad**. Ahora deberemos encontrar una solución mediante herramientas informáticas que permita, en mayor o menor medida, ayudar a la detección y tratamiento de estos problemas en los potenciales pacientes.

1.2 Contexto

Este proyecto se enmarca dentro del programa de becas colaborativas del **Instituto Valenciano de Investigación en Inteligencia Artificial** (de ahora en adelante VRAIN, por sus siglas en inglés). Dicho programa ha constituido las prácticas curriculares del autor que se han desarrollado durante el periodo comprendido entre noviembre de 2022 y junio de 2023. Durante estas, el alumno/autor se ha integrado dentro de un equipo multidisciplinar de investigadores denominado Grupo de Tecnologías Informática/Inteligencia Artificial (GTI-IA) cuya labor se centra (entre otras muchas), en el reconocimiento de las emociones, y su utilidad tanto en la detección como en la toma de decisiones con influencia de estas. Gracias a esto, se ha podido observar y conocer de primera mano el funcionamiento de las labores de investigación científica llevadas a cabo dentro de la universidad, así como ser partícipe en la toma de decisiones y ofrecer soluciones ante los dilemas planteados. En este caso, centrado en el apartado más directo con el control y monitoreo emocional de los posibles pacientes (el trabajo a desarrollar). El equipo de trabajo está configurado por más de diez miembros involucrados, habiendo presencia incluso de personal procedente de otras universidades nacionales mediante herramientas de trabajo en remoto (i.e. Microsoft Teams).

Otras tareas desarrolladas por el equipo interesantes de mencionar, es la combinación de **emociones y sistemas multiagente** (GenIA).

El objetivo de este área es obtener agentes con la capacidad de integrar emociones en su proceso de razonamiento. De esta forma, un agente ahora puede disponer de su propio **"estado emocional"** el cual puede variar a lo largo del tiempo, y usarse para su toma de decisiones. Esto otorga a los agentes un nuevo conjunto de creencias las cuales dotarán al agente de un comportamiento más "humano". Por ejemplo, un agente participe en un juego de apuestas como el póquer podrá tener asociado un estado emocional con respecto al transcurso y resultado de la partida. En el caso de realizar una jugada efectiva, el agente puede pasar a un estado "eufórico", lo que puede hacer que las acciones asociadas con apuestas más arriesgadas tengan ahora un mayor peso que en la instancia anterior. El funcionamiento interno del modelo se detalla en la figura 1.3

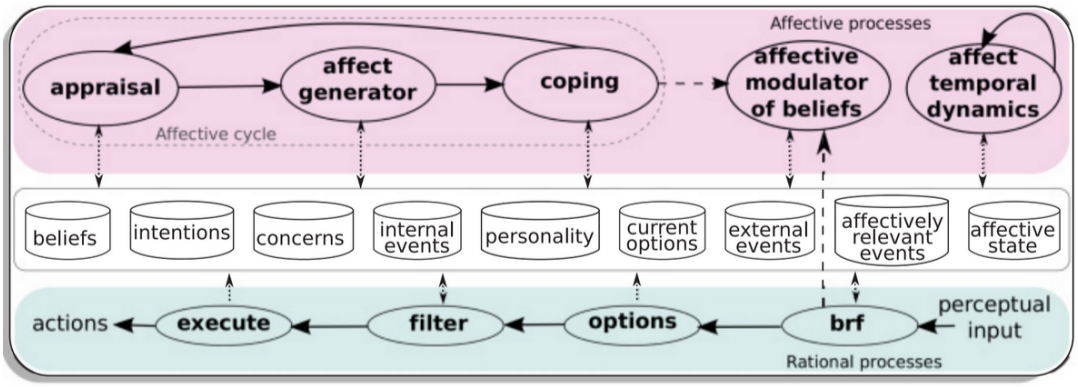

Figura 1.3: Arquitectura de GenIA. Fuente [3]

En el momento de escribir estas lineas, se ha aceptado en el congreso de **Practical Applications of Agents & Multi-agent systems** (PAAMS por sus siglas) un artículo describiendo la linea de investigación que proponemos es ente trabajo. Esto demuestra la relevancia que posee la línea de este trabajo, así como las otras partes de diferentes proyectos que convergen junto a esta en el mismo objetivo, dando un carácter muy ambicioso en cuanto a impacto esperado [15].

1.3 Objetivos

El objetivo principal de este trabajo es **desarrollar una herramienta de monitorización del estado emocional asociado a distintas actividades de personas ancianas**. Con ello, buscamos obtener un registro a largo plazo de la influencia en el estado afectivo del usuario ante diversas actividades que realice durante el transcurso de la jornada. Para lograr esto, hemos definido los siguientes subobjetivos que conforman la base del proyecto.

1. Crear una aplicación móvil de uso simple, económica y mínimamente intrusiva que detecte el estado emocional del usuario.

2. Definir e implementar un medio de persistencia de datos que permita obtener un registro histórico sobre el estado emocional del usuario.

3. Ofrecer una aplicación a los responsables que permita acceder a la información y generar informes sobre la situación afectiva histórica del usuario que tienen a cargo.

1.4 Impacto esperado

Tratándose este trabajo académico de una primera aproximación conceptual entre problema/solución, el resultado final y su impacto resulta de especulación abierta por parte del autor. Sin embargo, es importante siempre dar unas primeras estimaciones para entender el potencial de la solución a desarrollar.

Existen muchos escenarios de aplicación de la solución propuesta, entre los cuales podemos destacar los siguientes:

1.4.1. Profesionales de la salud mental

Debemos de destacar las labores llevadas a cabo por todos los facultativos del área de la salud mental y su importancia para el correcto tratamiento de los pacientes. Son ellos los que tendrán la última palabra respecto al estado y cuidados que este debe recibir así como el determinar su evolución. Por lo tanto, en ningún momento se busca sustituir las labores de diagnóstico mediante herramientas automatizadas. Siempre se ha de ver este trabajo como un punto de apoyo para estos.

Sin embargo, esto no impide que tareas más básicas y rutinarias se pueda automatizar hasta cierto punto. El monitoreo de los pacientes puede ser uno de ellos, ya que abarca muchas facetas de la vida diaria de una persona. Un caso típico es el de las reuniones periódicas entre médico/paciente que sirvan de punto de control para detectar cambios de conducta en este último. Podemos ayudar al facultativo en su tarea antes siquiera de ver al paciente, con un registro digital que permita ver de un rápido vistazo un historial de su estado emocional registrado en sus actividades diarias, además de combinarlo con técnicas y herramientas de análisis de datos.

Gracias a esto, podremos descargar de tareas rutinarias a los facultativos, mejorando las capacidades de desarrollo de su tarea médica permitiendo un mayor rendimiento y eficacia en el tratamiento de pacientes. A su vez, esto implica una menor dedicación para tratar al mismo paciente con el mismo resultado, y por consiguiente una reducción del tiempo en tratamiento, dando lugar a una mayor capacidad del facultativo de poder tratar un mayor número de pacientes. Con la misma plantilla de médicos disponibles en el sistema de salud, podemos **ampliar el número de pacientes** asumibles para el sistema.

Si consultamos el gasto real generado en un sistema de salud público como puede ser el catalán (véase la figura 1.4), podemos observar que las áreas de atención sociosanitaria y salud mental encajan perfectamente en el perfil tratado, y la suma total de su gasto puede alcanzar una cuota mayor al 25 %

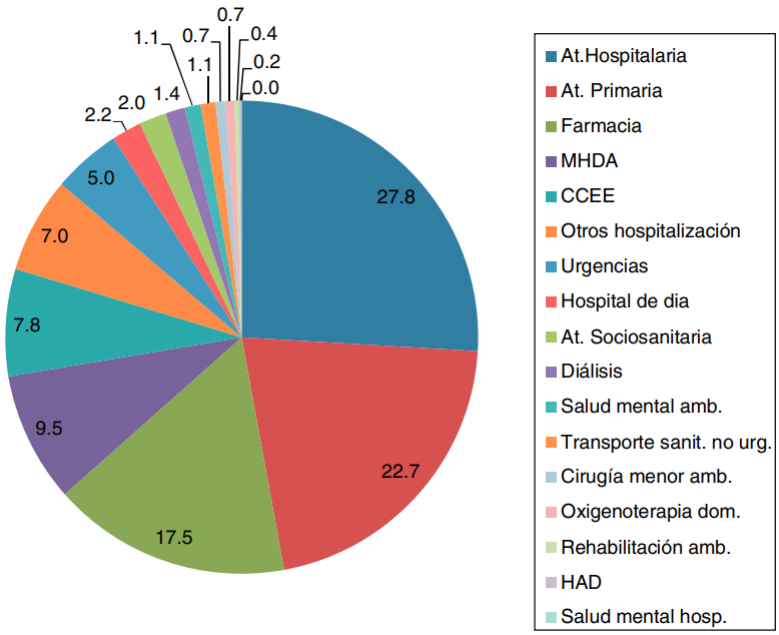

Figura 1.4: Distribución del gasto sanitario total por ámbito asistencial en Cataluña.
Fuente: [4]

1.4.2. Recopilación de datos representativos

La liberación de la aplicación al mercado y dentro de los diferentes sistemas sanitarios, generará un volumen de datos considerable y con un alto valor para el análisis y mejora del conocimiento de la evolución de los deterioros cognitivos asociados a la vejez. Estos datos pueden proceder de diferentes regiones geográficas con diferencias culturales y perfiles socio-económicos. Además, la generación de esta información será continua en el tiempo, y sujeta a los sucesos que acontezcan. Todo esto permite generar una base de conocimiento e información que se puede usar en nuevas investigaciones de carácter psicológico.

Debemos de tener muy en cuenta el tratamiento correcto de esta información, así como garantizar la ética en su uso, puesto que el tratamiento de esta se rige por las diferentes leyes y reglamentos de protección de datos que detallaremos en el apartado 2.4.

Esta aplicación podría también usarse a un nivel europeo gracias a las relaciones que existen entre las universidades, gobiernos, sistemas de salud y sistemas jurídicos. Además, forzando un poco las expectativas, estas aplicaciones a nivel internacional demuestran una mayor competitividad entre empresas/entes públicos. Esto es más notorio en los niveles de investigación (universidades), y se debe ver como un fomento en el interés por desarrollar tanto herramientas como simplemente ampliar el foco mediático respecto a las enfermedades mentales, y en especial al caso de soledad y el deterioro cognitivo-emocional en los mayores.

1.5 Estructura de la memoria

El resto de la memoria del trabajo se organiza de la siguiente manera:

El siguiente capítulo está dedicado al **Estado del arte** donde detallaremos cuál es el estado de la computación afectiva, su relevancia en el trabajo y las aplicaciones ya en el mercado con estas características. Además, daremos una explicación técnica sobre la representación dimensional y categórica de las emociones, una breve explicación de las tecnologías que vamos a emplear, para finalizar con un repaso a las cuestiones legales que debemos cumplir.

Seguidamente, en el capítulo titulado **Análisis del problema** presentaremos el diseño de la aplicación a desarrollar y sus requisitos. Para ello, debatiremos las funcionalidades que debe de aportar la solución base y los requisitos técnicos que debe cumplir.

El capítulo dedicado al **Diseño** detallará las decisiones tomadas para desarrollar la aplicación. Debatiremos las razones para la elección de la plataforma escogida (tanto *hardware* como *software*), así como los conceptos de las partes clave que componen la aplicación. Además, dedicaremos un apartado a explicar el desarrollo y funcionamiento del gestor de perfiles emocionales que hemos implementado.

Seguidamente el capítulo **Desarrollo e implementación** detallará el entorno de desarrollo que se requiere para el desarrollo/modificación del código creado, así como los componentes más técnicos de *React Native* para comprender su funcionamiento.

Finalizaremos esta memoria con las **Conclusiones y futuras ampliaciones** donde se expondrán las ideas clave que se han generado tras finalizar el trabajo. Además, detallaremos posibles ampliaciones que hemos considerado interesante

CAPÍTULO 2
Estado del arte

Antes de aventurarnos a dar una solución concreta al problema planteado, debemos analizar el estado actual de la materia, sus avances y productos ya disponibles en el mercado.

Debemos de tener en cuenta que la incorporación de razonamiento emocional en sistemas inteligentes es un área con un corto recorrido temporal. La gran expansión en cuanto a interés, inversión y popularidad de estas técnicas de inteligencia artificial y reconocimiento emocional se remonta a poco más de una década. Por ello, los avances que encontremos en el campo siempre se deben de tomar todavía como productos en desarrollo y poco maduros. Sin embargo, sería muy injusto plantear la **computación afectiva** de forma tan pesimista, pues durante los últimos años ha emergido un considerable número de proyectos que involucran las emociones de los usuarios finales, y la utilidad de los datos generados durante el proceso. Dicha información se encuentra en fase de auge, con numerosas empresas de renombre integrando soluciones que permitan conocer en tiempo real la reacción de los usuarios ante la información mostrada, de forma que esta pueda ser adaptada maximizando la atención y la sensación generada, mejorando por consiguiente la percepción del producto.

La multinacional **Coca-Cola** realizó una colaboración con el productor y DJ estadounidense *Marshmello* para la presentación de su nueva bebida donde emplearon reconocimiento de emociones para la experiencia. Gracias a ello, detectaban en tiempo real el estado emocional del usuario durante la visualización de la presentación [16].

2.1 Regulación emocional en aplicaciones móviles

Existe un término en psicología denominando **regulación emocional**, que consiste en la capacidad de un individuo de adaptar sus emociones ante eventos y el entornos en el que se encuentra. Estar regulación emocional también puede llevarse a cabo sobre otra persona, influenciando en el estado de ánimo de personas en un mismo entorno por acciones y comportamientos que nosotros mismos realizamos.

Ante el auge de los dispositivos móviles, se llevó a cabo un estudio que busca entender como las personas están empezando a usar sus dispositivos digitales como medio de regulación emocional [17]. Para ello, se creó una aplicación Android que empleaba la cámara frontal para registrar emociones con respecto a expresiones faciales. Dicha aplicación fue utilizada con 20 sujetos durante un periodo de 14 días a modo de experimento para detectar patrones no aleatorios, que pudieran indicar la existencia de regulación emocional.

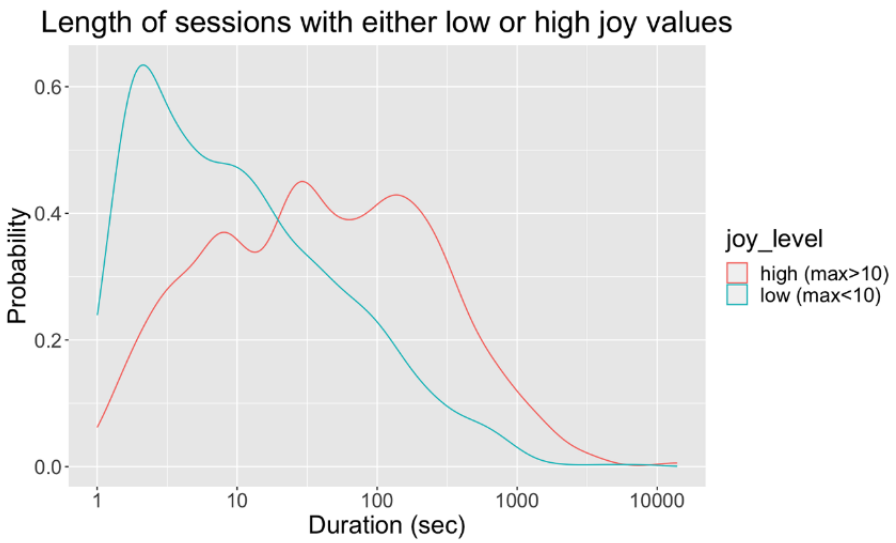

Figura 2.1: Probabilidad de duración respecto al uso del móvil

Mediante cálculos explicados en el citado artículo, obtuvieron la gráfica de la figura 2.1, donde se refleja la probabilidad de duración de la sesión con el dispositivo. Observamos dos variables independientes, representando un estado de disfrute elevado y reducido respectivamente. Como observamos, dicho nivel tiene relación con la duración de la sesión. Con niveles altos de disfrute, observamos un desplazamiento de la probabilidad desde duraciones más cortas de sesión, hacia más elevadas, indicando un mayor uso del dispositivo en estos casos.

Tras finalizar el experimento de 14 días, el equipo y los participantes tuvieron una reunión donde comentaron cuestiones relacionadas con el uso del móvil y sus emociones. El equipo concluyó con la siguiente idea:

«While participants showed a strong tendency to resort to smart-phones in response to negative emotions, the interviews yielded that positive emotions triggered mostly non-technology responses»[17]

Donde se identificaba una relación del uso del móvil con emociones negativas, no dándose este caso ante emociones positivas. Consideramos de gran importancia este estudio pues la regulación emocional desde estados negativos mediante el uso de dispositivos móviles puede ser de gran interés para futuras extensiones de nuestro trabajo.

Dentro del campo de la medicina, están apareciendo nuevas herramientas para el ciudadano que toman en especial consideración el concepto de **medicina personalizada**. Dichas herramientas aplica un enfoque médico que busca proporcionar tratamientos y cuidados de salud individualizados y adaptados a las características únicas de cada paciente. En lugar de aplicar enfoques generales para el diagnóstico y tratamiento de enfermedades, la medicina personalizada se basa en la comprensión de las características genéticas, moleculares y ambientales de un individuo para tomar decisiones clínicas más precisas.

La medicina personalizada puede utiliza avances en la genómica, la proteómica, la metabolómica y otras disciplinas científicas para obtener información detallada sobre la biología de un paciente. Esto implica el análisis de datos genéticos, afectivos, perfiles moleculares y factores de riesgo individuales para predecir la predisposición a enfermedades, diagnosticar enfermedades de manera más temprana y seleccionar el tratamiento más adecuado y efectivo. Existe un proyecto en estado de desarrollo que busca aplicar este concepto, y acercarlo a los ciudadanos de forma generalizada. Dicho proyecto es

llevado a cabo de la mano de los sistemas sanitarios de las comunidades autónomas de Canarias y Comunidad Valenciana, mediante financiación europea. Dicho proyecto tiene una finalidad asistencial y persigue mejorar la atención personalizada de pacientes, mediante el uso de herramientas tecnológicas punteras para ofrecer a la ciudadanía un servicio sanitario más avanzado y eficiente.

El objetivo de este proyecto es mejorar la atención personalizada (diagnóstico, tratamiento e investigación) de pacientes con enfermedades crónicas, oncológicas, degenerativas y raras, mediante herramientas de soporte a la decisión clínica. Este sistema estará basado en el procesamiento de grandes volúmenes de datos (o técnicamente denominado Big Data), en la atención directa y/o remota por profesionales de salud y/o sistemas alternativos de asistencia virtual.

Si bien este proyecto ofrece muchas características interesante, peca de una **interfaz poco adaptada a personal mayores**. Si observamos las capturas de la figura 2.2, cada función de la aplicación no se encuentra correctamente explicada y detallada, el tamaño del texto es inadecuado y su navegabilidad resulta confusa. Todo esto desemboca en la necesidad de un periodo de aprendizaje y adaptación al uso de la aplicación, lo que puede llevar a una desincentivación en su uso. Sobretodo, aquellos usuarios en edades más avanzadas que presentan dificultades ante el uso de estos dispositivos, puede ser una barrera de entrada muy grande para ellos, perdiendo la oportunidad de mejorar su calidad de vida por una interfaz no adaptada a ellos. Por esto, es muy importante que cuando desarrollemos nuestra solución tengamos en cuenta las necesidades específicas de nuestro público objetivo.

2.2 Marco teórico

En esta sección mostraremos los diferentes conceptos e ideas que serán necesarios para comprender los siguientes temas a tratar.

2.2.1. Reconocimiento y representación de emociones

Para monitorizar el estado emocional de los usuarios, es necesario disponer de técnicas que nos permitan reconocer las emociones, elegir un modelo de representación de estados emocionales y emplear sistemas de interpretación acorde a dichos modelos de representación.

Emoción

Vamos a comenzar identificando exactamente a qué nos referimos con el concepto de **emoción**. Podemos fácilmente buscar su definición consultando el diccionario de la Real Academia de la Lengua Española:

---

1. Alteración del ánimo intensa y pasajera, agradable o penosa, que va acompañada de cierta conmoción somática.

2. Interés, generalmente expectante, con que se participa en algo que está ocurriendo.

Las anteriores definiciones responden a ideas más propias del uso coloquial. Empleamos el término **emoción** para reacciones generalmente intensas (tanto positivas como negativas) ante ciertos eventos. Esta idea, realmente no es la mejor aproximación al concepto original, es una "licencia lingüística" empleada en las conversaciones más coloquiales. Buscando información en fuentes cercanas al campo de la psicología, encontramos la siguiente definición dada por el doctor *Juan Moisés de la Serna*:

«Las emociones son reacciones **psicofisiológicas** que representan modos de adaptación del individuo cuando percibe un objeto, persona, lugar, suceso o recuerdo importante. Psicológicamente, las emociones alteran la atención, hacen subir de rango ciertas conductas guía de respuestas del individuo y activan redes asociativas relevantes en la memoria.»[18].

De esta forma, las emociones se consagran como reacciones innatas para adaptarse a ciertos escenarios. Esto nos deja con una relación directa entre **contexto y emoción**, que nos permite un punto de partida donde buscarlas. Las diferentes situaciones cotidianas podrán tener como efecto o resultado, diferentes emociones.

Existe otro concepto muy importante, utilizado también en el día a día, que merece la pena pararse a explicar: los sentimientos. Muchas veces utilizamos los términos emoción y sentimiento de forma indistinta para referirnos al mismo concepto (generalmente a las primeras definiciones dadas por la RAE). Sin embargo, ambos conceptos aunque relacionados, identifican distintas ideas. Los sentimientos son **productos de las emociones**, parten como resultado de un conjunto de estas y se establecen durante largos periodos de tiempo. En cambio las emociones responden a estímulos concretos, dando respuestas cortas, concisas y de necesidad biológica para hacer frente a adversidades o necesidades encontradas. Con esto se quiere recalcar la importancia de entender la brevedad de las emociones. Son concretas y precisas, y generalmente los términos populares empleados hacen referencia errónea a estos, al no ser universales.

Representación de emociones

Necesitamos concretar la forma de representar información emocional, de forma objetiva y universal para todos los seres humanos. Para dar respuesta a esto, el psicólogo estadounidense *Paul Ekman* ideó un listado de 6 emociones básicas [19], citadas a continuación:

1. Sorpresa
2. Asco
3. Tristeza
4. Ira
5. Miedo
6. Alegría

Este nos brinda un conjunto de etiquetas que podemos emplear para la identificación de las emociones mediante la interacción con el paciente/usuario de forma directa, dada la fácil interpretación de estas.

Sin embargo, identificar las emociones tanto de una persona ajena, como las propias, resulta una tarea complicada. Todo aquello que se encuentre involucrado en las funciones psicológicas tiene intrínsecamente un sesgo, además de que puede existir falsedad y

ocultación del estado anímico genuino por parte del paciente (voluntariamente o no), o simplemente que los canales por los cuales tratamos de identificarlos no son fiables del todo (véase expresiones faciales, tono de voz, etc.).

Sería ideal encontrar alguna representación emocional basada en valores cuantitativos que hiciera más fácil el tratamiento de las emociones, pudiéndose relacionar con las etiquetas anteriores y obteniendo un conjunto de datos más fidedigno. Y es esto es lo que vamos a presentar a continuación mediante un sistema de representación multidimensional.

El campo del estudio dimensional de las emociones es bastante amplio, pero con aportaciones respaldadas y con la suficiente antigüedad para tomarse como punto de referencia en la representación. Un buen documento para comenzar a investigar es el artículo publicado en 1996 por parte de **Albert Mehrabian**, donde se describe un marco teórico para la medición de diferentes características fisiológicas que permitirán la clasificación emocional[20]. El artículo abre con la siguiente cita del propio autor (1980):

"An essential requirement of any integrated science is the availability of a few basic dimensions suitable for analyses of all its problems. Indeed, the fundamental difference between the natural and social sciences is that natural sciences have such basic dimensions (e.g., length, time, mass), whereas social sciences do not"

Como se puede observar, se recalca la importancia de tener un conjunto de dimensiones que permita caracterizar, en nuestro caso las emociones, evitando sesgos.

Mehrabian cita diferentes estudios y aproximaciones tanto suyas como de sus investigadores, dando como resultado una representación tridimensional, denominada **Pleasure-Arousal-Dominance** (PAD). Los campos hacen referencia a:

1. **Eje de valencia afectiva**: Corresponde al juicio cognitivo de evaluación, donde una evaluación alta del estímulo se asocia con un mayor placer inducido, y viceversa. Por ejemplo, excitación/relajación/amor en oposición a humillación/desinterés/-crueldad.

2. **Eje de activación**: Corresponde al nivel de intensidad de los cambios fisiológicos entre situaciones de tranquilidad y extrema activación. Por ejemplo, *tensión corporal/concentración/ejercitación* para valores altos de activación, o *inactividad/aburrimiento/relajación* para valores bajos.

3. **Eje de control**: Corresponde a la sensación de control e influencia sobre la situación actual y las personas involucradas. Por ejemplo, *enfado/audacia/poder* en contra de *ansiedad/miedo/soledad*

Estas características fisiológicas de las emociones son consideradas por el autor como observables, de forma porcentual, dándole a cada campo un rango numérico real. Cada valor representa la aproximación a los extremos del eje (por ejemplo, valores de -1 y 1 representar en el eje de valencia afectiva una situación totalmente desagradable o plenamente agradable, respectivamente). Podemos ver el conjunto de ejes como un mapa tridimensional, en la que existen regiones delimitadas que consideramos representan un estado afectivo estimado.

A pesar de que la teoría se basa en los tres ejes PAD (por sus siglas en inglés), lo cierto es que el uso de los dos primeros ofrece un nivel de discriminación en torno al 85 %, siendo crítico el uso del eje de control solo para los casos que comprendan la ansiedad y la hostilidad. Por ello, y por la dificultad que tenemos los humanos para asignar un valor a la dimensión de control, frecuentemente se emplean únicamente los dos primeros ejes (véase un ejemplo gráfico de la representación bidimensional en la figura 2.3. Además,

CRITERIOS	AUTORES Y EMOCIONES "BÁSICAS"
Afrontamiento	Arnold(1960); amor, aversión, desaliento, deseo, desesperación, esperanza, ira, miedo, odio, tristeza y valor.
Expresión facial	Ekman, Friesen y Ellsworth(1982); ira, alegría, miedo, asco, sorpresa y tristeza.
Procesamiento	Izard(1992); alegría, ansiedad, culpa, desprecio, asco, excitación, ira, miedo, sorpresa y vergüenza.
Relación con instintos	McDougall(1926); asombro, euforia, ira, miedo, asco, sometimiento y ternura.
Innatos	Mowrer(1960); dolor y placer.
Sin contenido proposicional	Oatley y Johnson-Laird(1987); felicidad, ira, miedo, asco y tristeza.
Adaptación biológica	Plutchik(1980); aceptación, alegría, expectación, ira, miedo, asco, sorpresa y tristeza.

Tabla 2.1: Criterios utilizados para diferenciar las emociones discretas [6]

una aproximación bidimensional es más sencilla de interpretar, y será más sencillo su tratamiento.

Si observamos la figura 2.4 representando el mapa emocional propuesto por Lang, Bradley y Cuthbert en 1999 [21], nos daremos cuenta de la aparición de un curioso patrón, una curva que simula la silueta de la letra **C**. Dicho patrón se puede explicar dado el hecho de que las ocurrencias de alta del eje de activación llevan consigo una valencia afectiva también en los extremos de este (tanto positiva como negativa). Esto determina que una valencia afectiva extrema, implica un alto nivel de activación, así como las situaciones de alta activación no suelen ser neutras en la valencia.

Los dos modelos que hemos detallado para la representación emocional (dimensional y discreta) presentan divergencias entre ellos. Los que apuestan por la primea, abogan por la necesidad de definir un mapa dimensional que permita detallar todas las posibles emociones, ya que las diferencias individuales en las emociones que desarrollan las personas excede la capacidad el sistema de etiquetas. Además, la representación dimensional nos ofrece un número "infinito" de estados emocionales, así como un esquema para delimitar las diferencias entre emociones.

Por otra parte, aquellos que optan el modelo discreto, defienden la existencia de emociones "primarias", teniendo características distintivas y únicas en sus elementos. Partiendo de esta, se desarrollarían las demás emociones secundarias. Para su clasificación, se emplean diferentes criterios según el tipo de emociones "básicas" que se quieran identificar. Estos criterios pueden ser las expresiones faciales, la adaptación biológica, la descarga nerviosa producida, etc. (véase la tabla 2.1 para más detalle).

2.3 Tecnologías

Cuando entremos en los apartados centrados puramente en el desarrollo del *software*, es importante conocer las tecnologías que se vayan a emplear, para comprender con profundidad qué se busca obtener en el producto final, y cómo es el funcionamiento de la

aplicación. El campo del desarrollo de *software* es conocido por su alto nivel de dinamismo y sus constantes actualizaciones. Las tecnologías cambian en cuestión de lustros, y es importante emplear las más recientes del momento, para obtener un producto lo más acorde a las necesidades y requisitos "impuestos" en la industria.

Como base del desarrollo, gran parte de la codificación (instrucciones generadas con el objetivo de ser ejecutadas por una computadora) se escribirá en el lenguaje interpretado **Javascript**. JS (por sus siglas) porta a sus espaldas años de asentamiento en la incorporación de la parte lógica en páginas web, siendo este uno de los lenguajes más empleados por los programadores en términos globales. Además, en los últimos años se ha podido apreciar su aplicación en entornos *backend* (toda la lógica detrás de "lo visual" en cuanto a aplicaciones se refiere), como puede ser el popular **Node.JS**. Todo ello, hacen de JS un lenguaje ya maduro y viable tanto para proyectos pequeños como de mayor envergadura.

2.3.1. React Native

Para la aplicación que deseamos desarrollar nos centraremos principalmente en dispositivos móviles. Las plataformas dominantes del momento (Android e iOS) permiten un trabajo sobre ellas con lenguajes asentados en la industria (Java en el primer caso, y Objective-C para el segundo), acompañados con una gran documentación y una extensa comunidad de desarrolladores. Sin embargo, crear aplicaciones para plataformas específicas requerirá siempre de un aprendizaje previo del uso de estas y de sus características propias que no suelen ser transversales hacia otras. Ante este problema, existen diferentes *fraemworks* (entornos de trabajo) que buscan una aproximación más fácil, agilizando la obtención de aplicaciones y buscando la portabilidad del código diseñado hacia diferentes plataformas. Un caso particular es el de **React Native**[1].

React Native nace de la mano de la empresa Meta (anteriormente conocida como Facebook) como un conjunto de herramientas y funciones dispuestas al programador para crear aplicaciones en múltiples plataformas (Android, Android TV, iOS, Windows, macOS y Web) basándose en el mismo código fuente con los cambios mínimos requeridos para cada plataforma.

La cualidad más importante es que las interfaces creadas (aquello con lo que el usuario interactuará) se basan en metodologías web, basadas en JavaScript y HTML. Esto permite crear aplicaciones nativas de igual forma y con la misma facilidad que crear páginas web. Tanto la creación de la interfaz de usuario, como la lógica detrás del procesamiento de la información, pasa por la codificación en lenguaje Javascript o Typescript (ambos prácticamente idénticos y combinables, con la única diferencia en el tipado de datos). Esto agiliza enormemente el desarrollo de aplicaciones y su posterior mantenimiento, y se ha postulado como un referente en cuanto a la tecnología a usar para el desarrollo de aplicaciones móviles multiplataforma a bajo coste y fáciles de mantener.

2.3.2. MorphCast

MorphCast[2] es una empresa italiana enfocada en ofrecer soluciones *software* de bajo coste y fáciles de implementar. Sus productos giran entorno al reconocimiento y *tracking* facial, obtener una serie de datos relevantes que permitan al cliente obtener un *feedback* sobre las emociones de sus usuarios, y su reacción durante su uso (como puede ser detectar la atención en una videoconferencia, la reacción ante la presentación de un nuevo

[1] Página oficial de React Native: https://reactnative.dev/
[2] Página oficial de MorphCast: https://www.morphcast.com/

producto, etc.). La empresa ofrece una API de muy bajo coste, tanto económico como computacional. Se basa enteramente en el desarrollo web (HTML/JS), por lo cual su uso es transversal para todas las plataformas, permitiendo además la integración con los *fraemworks* de desarrollo web más populares del momento (React, Vue, Angular, etc.)

Su uso se basa en la integración del uso de cámara durante el tiempo de ejecución de la aplicación, devolviendo al programador una serie de valores categorizados en módulos (atención, etiqueta emocional, posición facial, número de personas en la instancia, etc.), de entre los cuales puede seleccionar aquellos que le interese tener en cuenta. Toda la información se recaba mediante **eventos de Javascript**. Dichos eventos funcionan de la siguiente forma: El programador declara cómo tratará los diferente eventos y su información retornada, tras lo cual seguirá su ejecución con normalidad. Cuando se emita un evento, Javascript lo detectará automáticamente y será procesado en la forma que el programador detalló anteriormente, siguiendo la ejecución del programa una vez haya terminado el tratamiento del evento. Esta forma de ejecución se conoce como asíncrona, y es un pilar fundamental de Javascript, lo que lo convierte en la opción dominante en el mercado. En el caso del reconocimiento facial, esta aproximación simplifica enormemente la implementación, ya que toda la información es generada por el SDK, y ofrecida al programador de forma cómoda y formateada en estructuras estandarizadas (objetos de Javascript en este caso). De esta forma, solo nos deberemos de centrar en consumir la información relevante para nuestro caso, sin tener que implementar nada en específico y con una integración no-bloqueante en nuestro código.

2.3.3. Electron.JS

Electron[3] es otro *fraemwork* que nos ayudará a la hora de desarrollar aplicaciones de escritorio. Dicho *fraemwork* permite al programador utilizar el lenguaje **Javascript** para crear programas multiplataforma (Windows, macOS y Linux en sus diferentes versiones), de forma muy parecida a **React Native**. Para lograrlo, se identifican dos entornos de operación:

1. **Render** (renderizado), será el proceso/hilo encargado de todas las interacciones con el usuario (interfaz gráfica). Este proceso se escribirá también en Javascript mediante tecnologías web, de forma que la creación de estos es prácticamente idéntica a la creación de una aplicación web. Permite el uso de otros *fraemworks* web, como React.JS, Vue.JS o Angular.
2. **Main** es el hilo principal, encargado de atender todas las funcionalidades que se requieran. También se encuentra escrito en **Javascript**, empleando el entorno de Node, JS en este caso. Dicho *fraemwork* dota a JS de la funcionalidad *backend* típica de una aplicación. En el caso de Electron su uso es muy superficial, y no requiere de profundizar mucho en la materia. Basta con entender que toda la lógica respecto al tratamiento de datos pasa por este punto)

La importancia de tener dos entornos separados radica en la seguridad que esto proporciona. Toda la interacción con el usuario (y por lo tanto el tratamiento de datos) se encuentra aislada del proceso *main* y la comunicación entre ellos se produce mediante APIs bien definidas y acotadas por parte del programador. Esto evita el peligro de un acceso no autorizado a funcionalidades del sistema operativo por algún ataque producido en el lado del usuario. Por ejemplo, la selección de archivos de disco se debe de definir al crear la aplicación, lo que permite al programador limitar su acceso e impedir que terceros tengan un acceso directo a estos.

[3]Página oficial de Electron.JS: https://www.electronjs.org/es/

Express.JS[4] es un entorno de trabajo pensado para ser ejecutado dentro de Node.JS [5], con la capacidad de brindar funcionalidades API e intercambio de información mediante diferentes protocolos web, como HTTP. Gracias a Express, podremos crear de forma rápida y muy sencilla diferentes *endpoints* (puntos finales de comunicación). Un *endpoint* es una ruta específica donde interactuaremos de forma directa con la aplicación, y sus funcionalidades de tipo **CRUD** (Create, Read, Update y Delete).

2.4 Marco legal

Cuando el usuario emplee la aplicación, se generará una gran cantidad de datos personales. No solo los que pueda introducir de forma explícita en referencia a su estado de ánimo, si no además todos los registros faciales obtenidos y la información afectiva derivada de estos. Toda información personal de los usuarios ha de ser tratada según la legislación vigente, que requiere la necesidad de asegurar la información ante accesos no autorizados.

2.4.1. Unión europea

La carta de los derechos fundamentales de la Unión Europea recoge, entre otros muchos, el derecho de los ciudadanos a que se proteja y limite el acceso de sus datos personales. El reglamento que regula esta cuestión es el RGPD 2016/679 [22] del parlamento europeo y del consejo (27 de abril de 2016), el cual se encuentra actualmente en vigor y define los **datos personales** como:

«toda información sobre una persona física identificada o identificable («el interesado»); se considerará persona física identificable toda persona cuya identidad pueda determinarse, directa o indirectamente, en particular mediante un identificador, como por ejemplo un nombre, un número de identificación, datos de localización, un identificador en línea o uno o varios elementos propios de la identidad física, fisiológica, genética, psíquica, económica, cultural o social de dicha persona»

De igual forma, define el **tratamiento** de estos como:

«cualquier operación o conjunto de operaciones realizadas sobre datos personales o conjuntos de datos personales, ya sea por procedimientos automatizados o no, como la recogida, registro, organización, estructuración, conservación, adaptación o modificación, extracción, consulta, utilización, comunicación por transmisión, difusión o cualquier otra forma de habilitación de acceso, cotejo o interconexión, limitación, supresión o destrucción»

Definidos los conceptos de datos personales y tratamiento, debemos tener en consideración los límites establecidos por la UE para considerar un tratamiento de datos personales lícito. Para que el tratamiento se considere lícito, debe de cumplir como mínimo uno de los siguientes casos:

1. El interesado dio su consentimiento para el tratamiento de sus datos personales para uno o varios fines específicos.
2. El tratamiento es necesario para la ejecución de un contrato en el que el interesado es parte o para la aplicación a petición de este de medidas precontractuales.
3. El tratamiento es necesario para el cumplimiento de una obligación legal aplicable al responsable del tratamiento.

[4] Página oficial de Express.JS: https://expressjs.com/es/

[5] Node.JS (https://nodejs.org/es) es un entorno de ejecución basado en Javascript que responde a las necesidades de implementar funcionalidad de servidos en aplicaciones JS. No vamos a profundizar más en esta herramienta, pues no se emplea de forma directa en el trabajo

4. El tratamiento es necesario para proteger intereses vitales del interesado o de otra persona física.

5. El tratamiento es necesario para el cumplimiento de una misión realizada en interés público o en el ejercicio de poderes públicos conferidos al responsable del tratamiento.

6. El tratamiento es necesario para la satisfacción de intereses legítimos perseguidos por el responsable del tratamiento o por un tercero, siempre que sobre dichos intereses no prevalezcan los intereses o los derechos y libertades fundamentales del interesado que requieran la protección de datos personales, en particular cuando el interesado sea un niño.

En nuestra aplicación, debemos optar por obtener de forma explícita el consentimiento del usuario para recopilar la información tanto generada en la interacción, como en segundo plano. Debemos detallar qué uso se le dará, las limitaciones de esta y su medio de persistencia.

2.4.2. Legislación española

La justicia española se rige por la **Ley Orgánica de Protección de Datos Personales y Garantía de los Derechos Digitales** (LOPDGDD) 03/2018, aprobada el 5 de diciembre de 2018 y publicada en el BOE número 294 el 6 de diciembre del mismo año [23]. Dicha ley aplica al territorio nacional (España) y refuerza la correcta aplicación del reglamento europeo y su ajuste a las características propias del sistema judicial español, tal y como enuncia en el artículo 1:

1. Adaptar el ordenamiento jurídico español al Reglamento (UE) 2016/679 del Parlamento Europeo y el Consejo, de 27 de abril de 2016, relativo a la protección de las personas físicas en lo que respecta al tratamiento de sus datos personales y a la libre circulación de estos datos, y completar sus disposiciones.

2. El derecho fundamental de las personas físicas a la protección de datos personales, amparado por el artículo 18.4 de la Constitución, se ejercerá con arreglo a lo establecido en el Reglamento (UE) 2016/679 y en esta ley orgánica.

3. Garantizar los derechos digitales de la ciudadanía conforme al mandato establecido en el artículo 18.4 de la Constitución.

Esta ley sigue las directrices impuestas por el RGPD y no tiene ningún apartado específico que añada nuevas obligaciones al tratamiento que debamos realizar.

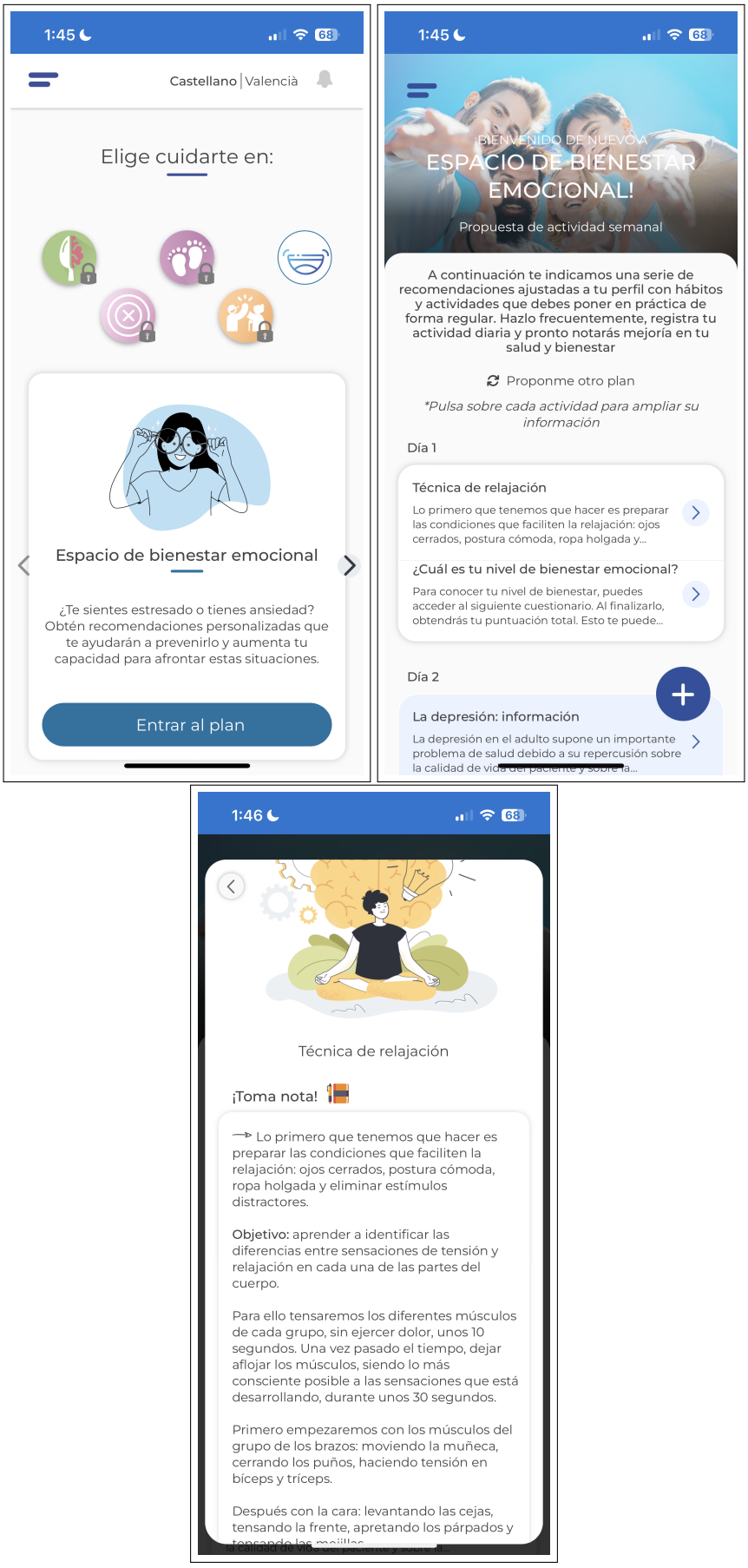

Figura 2.2: Capturas de la interfaz Ciudat-e

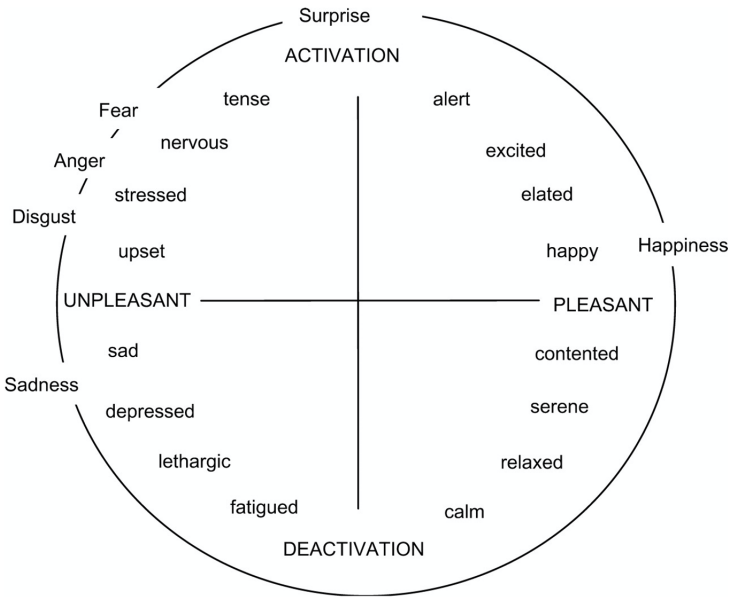

Figura 2.3: Espacio circunflejo propuesto por Rusell. Fuente [5]

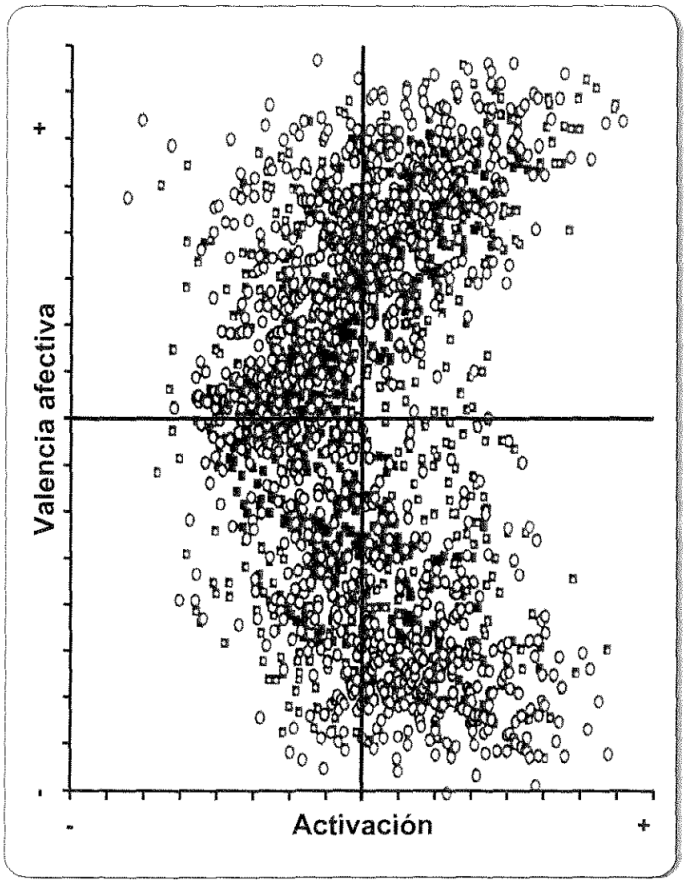

Figura 2.4: Espacio emocional bidimensional. Fuente [6].

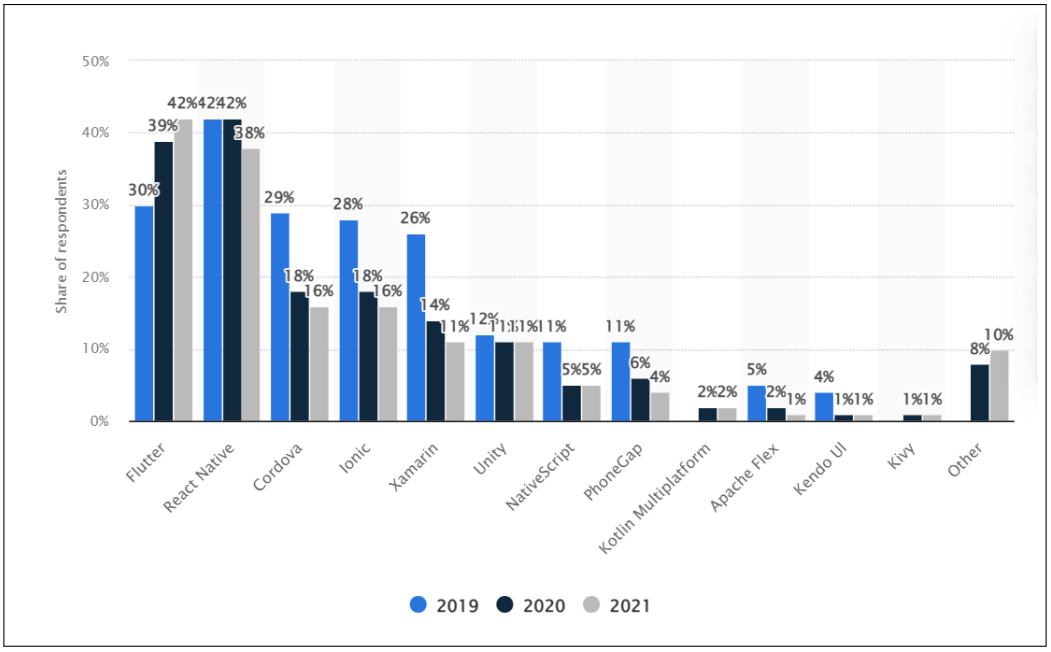

Figura 2.5: Porcentaje de aplicaciones creadas con los diferentes *fraemworks*. Fuente [7]

CAPÍTULO 3
Análisis del problema

Vamos a definir de una forma concreta qué se le ofrecerá al usuario en una primera versión de la aplicación. Para ello, listaremos las funcionalidades que debe ofrecer y con qué objetivo. Su diseño e implementación se detallarán más profundamente en futuros apartados.

3.1 Funcionalidades base

Recordemos las términos claves que venimos usando hasta el momento: **contexto, emoción y reconocimiento**. Queremos obtener relaciones entre las situaciones o actividades del usuario con posibles respuestas emocionales ante ellas. Además, vamos a tratar de hallar una correlación entre estas con registros obtenidos de las expresiones faciales registradas del usuario, mediante la teoría mostrada en el apartado 2.2.1. Por ello, el funcionamiento de la aplicación girará entorno a tres escenarios descritos a continuación

3.1.1. Agenda emocional

El principal problema a enfrentar es la sensación de soledad y aislamiento que afectan al usuario, así que hemos de tener muy en mente siempre la interacción con el usuario. Esto no quiere decir que la solución deba aportar un extenso abanico de funcionalidades y opciones, pues el público objetivo será mayoritariamente de personas en edades avanzadas. Dichas personas, por norma general, disponen de más limitaciones en el campo tecnológico, por lo cual una solución muy abrumadora desincentivaría el uso de la aplicación. Deberemos de tratar de ofrecer una interacción con el usuario lo más limpia, sencilla y visual para el usuario. La interfaz con la que interactúe el usuario tiene que permitir un uso ágil, rápido y ameno, de forma que su introducción en la vida cotidiana sea lo más sencillo posible.

Hay que recalcar que el objetivo final del proyecto es recabar un amplio volumen de información en periodos temporales extensos, por lo cual la aplicación debe de tener un uso diario durante largos periodos de tiempo sin interrupción. No obstante, se puede tolerar la falta de información para un pequeño número de días sin interacción). De esta forma, los datos que obtengamos serán de corta información pero precisa, y obtenidos directamente al usuario, para mantener el contacto con este.

La información recopilada será de la forma **Contexto/Emoción**, donde el usuario asocia una situación o actividad con un estado emocional inicial y final.

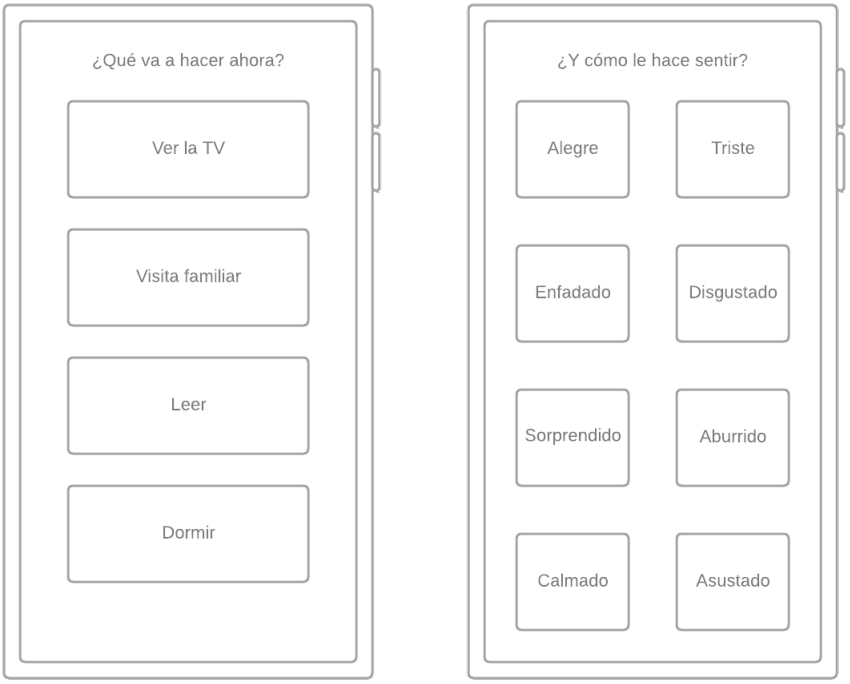

Figura 3.1: Ejemplo conceptual aplicación móvil

3.1.2. Reconocimiento facial

Una parte muy interesante de este proyecto es la de recabar información anímica mediante el análisis de gestos faciales, que permitan obtener una serie de valores numéricos los cuales permitan su tratamiento posterior, análisis y predicciones. Es importante resaltar el hecho de que no se busca una identificación de *etiquetas* emocionales. No esperamos trasladar los vectores obtenidos a un conjunto de etiquetas específicas *(i.e. Alegría, Miedo, Tristeza, etc.)*, si no obtener una representación de la emoción basada en sus componentes universales: Pleasure y Arousal, que permitan identificar patrones considerados como **situación normal/estable**, y distinguir cuando el usuario se encuentre anímicamente alterado (un pico de activación tras una noticia impactante, una baja activación prolongada que represente apatía, etc.). Además, la emoción reconocida en el rostro del usuario se comparará con la emoción indicada por el usuario en un proceso de autoevaluación emocional recogido por la aplicación.

3.1.3. Medio de contacto

Si podemos registrar las emociones del usuario y obtener una representación fiel sobre su carácter, tendremos la capacidad de detectar situaciones anómalas, determinar si estas son perjudiciales o no deseables para el usuario, e iniciar un proceso de comunicación tanto con el usuario, como con las partes necesarias para cuidar del paciente. Por cada paciente se definen 3 personas de contacto en función del nivel de alerta detectado por la aplicación:

1. Familiar o persona cercana a cargo del usuario
2. Profesional sanitario
3. Servicio de emergencia (112 por ejemplo).

Una interacción básica, pero cercana ayuda al usuario a no sentirse completamente solo, sentir que *alguien* está observándole, está preocupado y se ha percatado de que no se encuentra bien. A partir de aquí, las opciones son muy amplias y existen muchos ejemplos en los que basarse. Una opción sería dotar a la solución de un *avatar* visualmente amigable, al estilo de los populares *Miis* de Nintendo (véase la figura 3.2), que interactúe mediante *text to speech*, y que le consulte al usuario sobre como está, que le preocupa, o distraerlo contándole una historieta, chistes e incluso una pequeña conversación intrascendente que le desvíe la atención del problema original. Esta funcionalidad queda fuera del alcance de este trabajo, pero creemos que debería formar parte de la solución global que se está desarrollando a largo plazo.

Figura 3.2: Ejemplo conceptual, mostrando la aplicación de Miitomo (Nintendo) que permite la conversación entre usuario/avatar. Fuente [8]

3.2 Requisitos

Para lograr obtener una aplicación que responda a los objetivos descritos, debemos listar qué requisitos y funcionalidades debe de poseer nuestra solución final, a modo de guión de trabajo. Los principales son: **Modular, uso sencillo, consumo mínimo de recursos, bajo coste e intrusión mínima**. A continuación se detallan cada uno de estos.

Modular. Trabajamos en ofrecer una aplicación que responda ante las necesidades descritas del usuario. Dichas necesidades pueden ser varias, como las descritas anteriormente de monitoreo anímico y canal de aviso para los familiares y profesionales de la salud. Además, recordemos que este trabajo se encuentra integrado dentro del área de investigación del VRAIN (detallado en el capítulo 1.2), donde existen varias ramas de desarrollo independiente, pero con el mismo objetivo final: atender las necesidades de las personas vulnerables a la soledad. Esto hace tomar una forma **modular** a la aplicación. Dicho modelo consiste en la creación de una aplicación

principal o base, con unas funcionalidades bien delimitada. A esta base, se le podrá añadir nuevas de forma sencilla, dado que no hará falta reescribir nada del código para poder integrarlas. De esta forma, la aplicación ofrecerá al usuario características bien definidas y operativas, con opción a ampliarse de cara al futuro según las necesidades surgidas, o las nuevas ideas surgidas en el equipo.

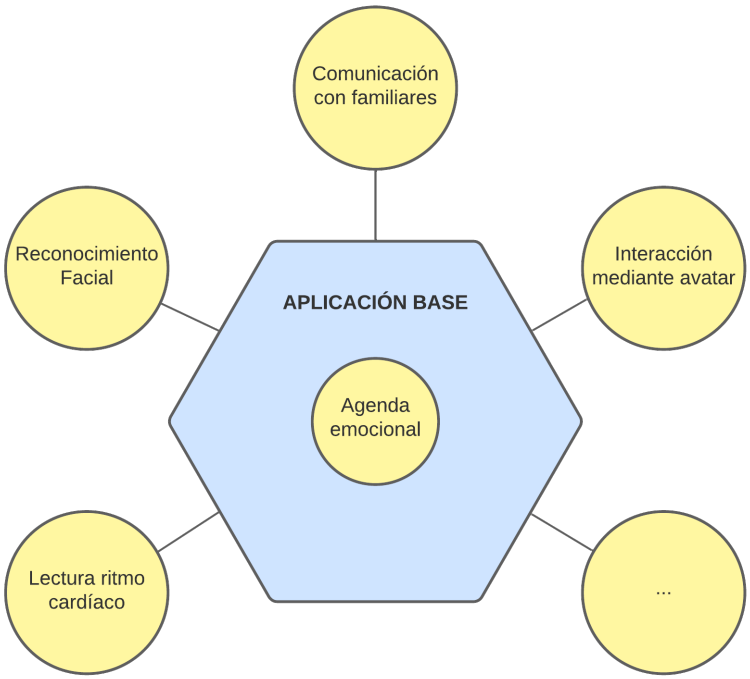

Figura 3.3: Diagrama arquitectura modular

Uso sencillo. La aplicación estará abierta a todo el público. Sin embargo, esta claro que el enfoque principal será el de las personas en edades avanzadas. Dicho grupo puede no tener las habilidades o destrezas de aquellos que han crecido envueltos en toda clase de dispositivos electrónicos, así que es importante adaptar la experiencia de uso. Esto tiene ciertas implicaciones y/o limitaciones a la hora de entregar una interfaz gráfica con la cual puedan interactuar. Ofrecer muchas opciones y funcionalidades con las que interactuar pueden abrumar al usuario, haciendo que su uso resulte complejo, denso y que requiera un cierto periodo de adaptación y aprendizaje. Esta barrera con el usuario puede ser decisiva en la continuación de su uso, ya que la primera toma de contacto con la aplicación y las sensaciones generadas en esta determinarán de forma notable el interés del usuario con la aplicación.

Consumo mínimo de recursos Integrar la aplicación en el entorno de los *smartphones* implica abarcar un número considerablemente grande de dispositivos en el mercado, con sus características únicas, que responden a las necesidades únicas de cada usuario (habrán aquellos que requieran más prestaciones, y otros los cuales les basta con poder realizar las actividades más básicas como consultar el correo y enviar pocos mensajes). Además, la introducción y popularización de estos se remonta a principios de la década de 2010, por lo cual tenemos que tener en cuenta que existirán usuarios que empleen de forma cotidiana dispositivos con características técnicas antiguas, con las limitaciones que ello conlleva.

También deberemos tener en cuenta el hecho de que en muchas familias, aquellos que realicen un uso más intensivo de estos, actualizan sus dispositivos de forma

más recurrente, y delegan o "regalan" estos a sus mayores. Este fenómeno se suele dar en familiares más jóvenes que el usuario a tratar; actualizan sus dispositivos para obtener las nuevas características desarrolladas en la industria, pero sus dispositivos actuales son perfectamente utilizables para tareas más sencillas, como son las realizadas por sus mayores. Esto implica que los ancianos no suelen adquirir dispositivos nuevos, y emplean aquellos que van por detrás en cuanto a novedades móviles se refiere.

Con esta puesta en contexto, se nos presenta la necesidad de ofrecer una aplicación que consuma pocos recursos y sea funcional en dispositivos con versiones antiguas de APIs, SDKs y sistemas operativos para poder alcanzar a nuestros potenciales usuarios, sin la necesidad de adquirir un dispositivo más moderno o con prestaciones elevadas.

Bajo coste Si continuamos con el anterior apartado, podemos ver que dándole un perfil bajo en cuanto a consumo de recursos, se permite el uso de dispositivos de bajas prestaciones. Esto es muy importante, puesto que permite a usuarios con un poder adquisitivo más limitado acceder a esta, mediante la "herencia" de dispositivos por familiares, o la posibilidad de adquirir aquellos que se presentan como gama baja, los cuales presentan un coste mucho más accesible que dispositivos de gama media-alta. Con esto podremos evitar generar una desigualdad en el uso de la aplicación y hacerla accesible para todas las personas. Incluso se puede ver un beneficio de esto en los centros sociosanitarios encargados del cuidado de un gran número de pacientes. Evitar la necesidad de una gran inversión en dispositivos para los residentes, animará a estos centros a incluir nuestra solución en su programa de atención y cuidados.

Intrusión mínima El objetivo de la aplicación es su uso continuado en el tiempo, integrándose en el día a día del usuario. La interacción entre ambas partes debe ser posible de la forma más rápida y automática posible: si el hacer uso de esta implica una larga pausa antes o durante la realización de la actividad, su uso resultará de gran molestia, reduciendo la probabilidad de que el usuario la integre en su vida. Esto lleva a la necesidad de ofrecer un flujo de uso de la aplicación lo más rápido, guiado e intuitivo posible. Esto, junto a lo explicado en el apartado 3.2, nos obliga a ofrecer de cara al usuario el número mínimo de operaciones que este pueda hacer. Esto no afecta a otras funcionalidades como pueda ser la comunicación con los familiares, el registro de emociones por cámara u otras más, ya que estas no interactúan de forma directa con el usuario, ni se dan en todos los usos que este haga.

CAPÍTULO 4
Diseño

4.1 Arquitectura

Debemos acotar las plataformas, herramientas y entornos en los que trabajaremos, pues es imposible crear una aplicación que abarque todos los medios digitales. A continuación, se detallará las decisiones finales antes de dar paso a la creación en si de la aplicación.

4.1.1. Plataforma *hardware*

Como mencionamos en el apartado 1.3, nuestra solución ha de ser lo más austera y económica de implementar. Esto pasa por prescindir de cualquier herramienta específica para el análisis cuantitativo emocional, así como el uso de cualquier tecnología de terceros que implique el pago de tasas o adquisiciones de servicios. Por lo tanto, vamos a idear un proyecto basado al cien por cien en la entrega de un producto **software**, además de que dicho desarrollo deberá basarse en tecnologías de coste cero (i.e. *open source*).

Para el uso de nuestro producto *software*, sigue siendo necesario una plataforma *hardware* que permita ejecutarla. Hemos de determinar que plataforma es la más económica y/o con mayor penetración en el mercado objetivo, y que tenga los útiles o periféricos necesarios para nuestra tarea. El dispositivo que mejor cumple estos requisitos es el **smartphone** o teléfono inteligente. De entre todos los dispositivos computacionales, es posiblemente el que ofrezca la interfaz más fácil e intuitiva. Ideal en nuestro caso.

Las interfaces entregadas por los sistemas operativos mayoritarios en los que se basan estos dispositivos (i.e. Android e iOS) son increíblemente fáciles de utilizar, con una interacción basada en la simplicidad del uso, elementos visuales y con el mejor dispositivo de interfaz humana, **los dedos de la mano**. Parece todavía más adecuado para nuestro contexto, en el que los principales usuarios serian personas en edad avanzada. Sin querer dejar entrever que los mayores tengan menos capacidades intelectuales que los demás grupos etáreos, es evidente que la aparición de computadoras personales en la forma masiva que hemos vivido los más jóvenes, no se dio en su época. Su aparición, para ellos tardía, implica la necesidad de aprender terminologías y conceptos totalmente ajenos a ellos. Unido además, al hecho de que es normal a estas edades haberse reducido las capacidades de aprendizaje.

Sin embargo, el caso de los teléfonos inteligentes ha supuesto un acercamiento máximo de la computación personal a este grupo. Su facilidad de uso, unido a los grandes beneficios de comunicación y ocio han hecho que, a día de hoy, sea habitual ver a personas mayores disponer de su propio dispositivo personal al cual se encuentran, en mayor

o menos medida, acostumbrados a su uso. Como ejemplo, podemos ver el crecimiento del uso de redes sociales ligadas a estos dispositivos en los grupos elevados de edad, como puede verse en la figura 4.1. Además, estos dispositivos se encuentran equipados con un gran número de diferentes sensores y periféricos, que facilitan la toma de información relevante por parte del usuario mediante **teclados virtuales** o la interacción con **botones**. En nuestro caso, emplearemos también la cámara frontal, presente en todos los dispositivos móviles para la toma de muestras faciales, en pos de nuestro objetivo de obtener valores cuantitativos.

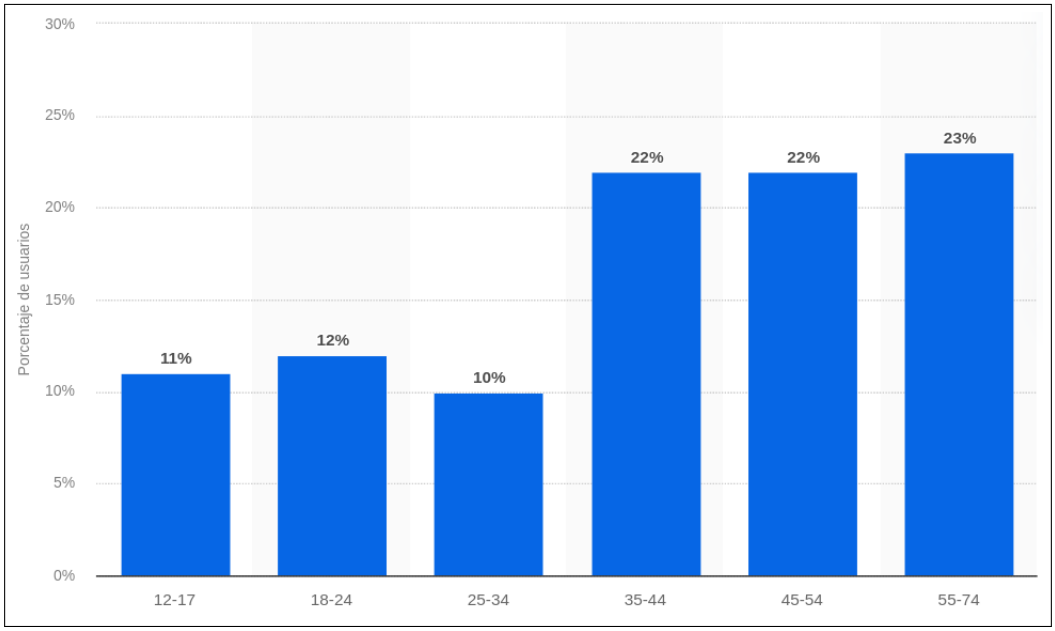

Figura 4.1: Porcentaje de usuario de redes sociales en España. Año 2022. Fuente:[9]

4.1.2. Plataforma *software*

Ya tenemos identificada la plataforma *hardware* donde ejecutaremos nuestra aplicación. Ahora necesitamos lo más importante: la aplicación *per se*. Escoger en que entorno vamos a desarrollar nuestro código es tan importante como la plataforma *hardware* de trabajo. Será esta la que determinará la forma en la que trabajemos, las herramientas que utilizaremos para la persistencia/interfaz/comunicación y el despliegue posterior en las plataformas para el acceso al público.

Vamos a comenzar definiendo los dos principales sistemas operativos existentes a día de hoy: **Android** e **iOS**. Si bien existen otros en el mercado, la cuota de los dos anteriores suma el 95 % del total. Es evidente que en un trabajo académico de este tamaño nos centraremos en estos dos. Por desgracia, también nos tocará descartar el desarrollo para la plataforma de Apple. Para la creación de aplicaciones en esta, es necesario disponer tanto de un dispositivo final para *testeo* (iPhone), como un computador de la familia *Mac* que permita ejecutar el sistema operativo macOS para la compilación [1]. Existe una desventaja final bastante importante en el caso de que se llegara a una distribución al público de la aplicación: El coste para publicar una aplicación en la tienda oficial de Android (Google Play) es de 20$ en concepto de adquisición de una licencia de desarrollador. Dicha pago

[1] Hace falta recalcar que esto no es del todo cierto. Existe la plataforma Expo que permiten la compilación de código para plataforma iOS desde otros dispositivos que no sean Apple, pero su uso tiene ciertas condiciones que descartan su uso. Además, sigue siendo requisito necesario el disponer de un dispositivo iPhone para las pruebas pertinentes

es único, y permite publicar tantas aplicaciones como el programador desee. En el caso de Apple, el coste de publicación en su tienda (App Store) es de unos 99$ en concepto de pago anual. Si bien en una puesta en marcha de la aplicación, se podría publicar la aplicación también en el mercado de Apple, decidimos centrarnos en la plataforma de Google para el desarrollo de este trabajo, pues la cuota de mercado de Android llega a superar el 80 %, lo que nos da un gran público (véase las gráfica históricas sobre la cuota de mercado de cada sistema operativo 4.2).

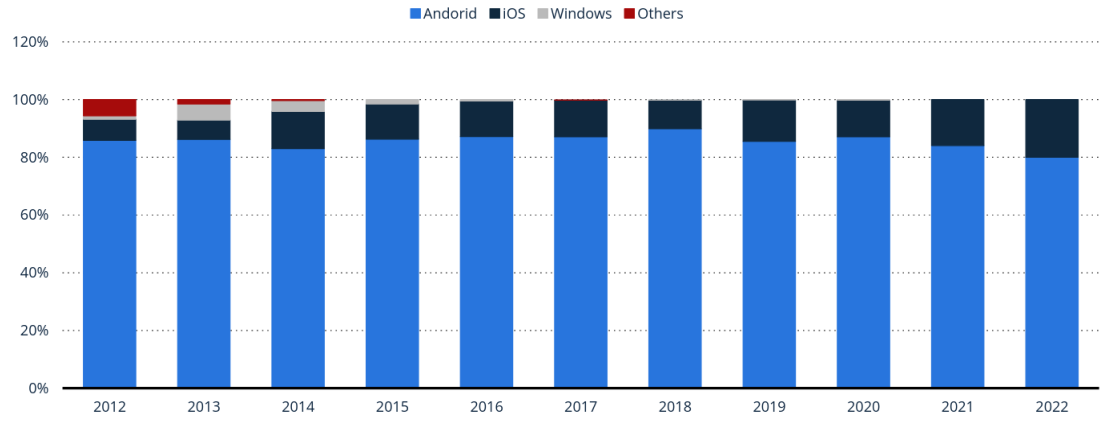

Figura 4.2: Porcentaje de usuarios para los diferentes OS. Fuente [10]

Teniendo claro que el desarrollo será exclusivo para **Android**, podemos comenzar a trabajar sobre dos plataformas de desarrollo. La primera opción es la de usar el propio entorno que nos ofrece el ecosistema. Dicho entorno consiste el la creación de código Java/Kotlin nativo, e interfaces mediante manipulación de código XML. Todo ello, sobre el IDE (*integrated development environment*) **Android Studio**, que no es más que una capa de personalización sobre el IDE **IntelliJ IDEA**, de la empresa JetBrains. Desarrollar una aplicación de esta forma otorga al programador control total sobre el comportamiento del código y acceso directo a funcionalidades del sistema operativo. Como contrapartida, crear una aplicación desde cero en este entorno resulta bastante tedioso; requiere de conocimientos previos sobre su funcionamiento, puesta en marcha y operabilidad antes siquiera de empezar a desarrollar cualquier aspecto de la aplicación.

Otra opción que ha ganado mucho peso y popularidad entre los desarrolladores, es el uso de entornos de trabajo enfocados en el uso de **Javascript**. Existen muchos de estos, que abstraen al programador de las tareas de más bajo nivel (hablando en términos de *hardware*), y le permiten enfocarse en la implementación de las funcionalidades deseadas. Estas aproximaciones tratan de integrar las metodologías del desarrollo web en aplicaciones móviles, lo que permite una mayor facilidad y rapidez en el trabajo. Esto no quiere decir necesariamente que sea una solución más básica o con peores prestaciones. En los casos de pocos integrantes en un proyecto, contar con estar herramientas es clave para poder ofrecer el producto en plazos de tiempo razonables, así como ser viable el mantenimiento posterior. El uso del entorno nativo se reserva a los casos donde sea imprescindible características muy concretas del dispositivo, o se trate de una aplicación con un gran equipo detrás ya que, por norma general, el rendimiento ofrecido suele ser superior que en los casos que se usen entornos de trabajo **JS**. En nuestro caso, el desarrollo se llevará a cabo íntegramente por un solo miembro. Por lo cual, resulta muy conveniente el uso de los entornos de trabajo anteriormente descritos.

Existe múltiples ofertas en cuanto a entornos ya creados y respaldados tanto por empresas de renombre como por la comunidad. Podemos seguir un simple criterio de popularidad para escoger nuestro entorno. Esto suele garantizar que el entorno se encuentre respaldado por una amplia comunidad y con una extensa documentación. Observando

la figura 2.5, podemos observar que durante los últimos años el referente ha sido **React Native** (si bien *Flutter* parece haberle ganado terreno igualándolo), siendo esta la escogida finalmente para la creación de la aplicación final (además de por el criterio personal de autor).

4.2 Diseño de la app

4.2.1. Sesiones

Antes de comenzar a explicar el funcionamiento de la aplicación, debemos de definir un concepto propio que emplearemos en el trabajo restante. Las **sesiones**. Las sesiones son instancias de la aplicación que contendrán los valores de **contexto** y **emoción** introducidos por el usuario, así como la fecha y hora en la que se registró. Dichas instancias se almacenarán en la base de datos, e irán identificadas por un valor numérico único, y clasificadas por su fecha. Las sesiones realizarán la función de puesta en común de cualquier otro módulo de la aplicación. Los registro vectoriales obtenidos del análisis facial (por ejemplo) estarán asociadas dentro de la sesión que se encontrara en marcha al momento de obtenerlos. Este funcionamiento se replicará a los nuevos dispositivos de reconocimiento que pudieran añadirse (como el análisis del ritmo cardíaco 6.1). De esta forma, será muy sencillo realizar un análisis del sujeto a posterior, y detectar la evolución de este.

4.2.2. Feedback

Dentro del apartado de agenda emocional, vemos tan importante como se siente el sujeto tanto al comenzar la actividad como las sensaciones tras finalizar. El contraste entre las **expectativas** antes y tras, permitirán analizar como el ánimo del sujeto varía, y detectando patrones entre diferentes emociones que lleven a un mismo escenario. Por ejemplo, si el conjunto de actividades a desarrollar en la calle tiene como resultado un estado emocional agitado, ansioso y deprimido, podríamos interpretarlo como un indicio de **agorafobia** (Aparición de síntomas propios de la ansiedad al encontrarse en espacios abiertos, y/o donde no se disponga de ayuda). Por ello, establecemos que una sesión no se encuentra finalizada hasta que el usuario no nos ha proporcionado la retroalimentación necesaria. Entonces, las sesiones se consolidan como dos pares contexto/emoción, diferenciados por el periodo temporal invertido en la realización de la actividad.

4.2.3. Flujo de uso

Definiremos de forma concreta el **flujo de uso** de la aplicación. Como mencionamos en el apartado 3.2, la aplicación debe de ser fácil de usar, sin complicaciones y con cero intromisión en el uso cotidiano del móvil por parte del usuario. De esta forma, hemos definido el flujo mostrado en la figura 4.3. La forma de uso es la siguiente:

1. El usuario accede a la aplicación, de la forma que el prefiera (Acceso directo desde el escritorio, relanzar desde las aplicaciones en segundo plano, etc.)

2. Directamente, la aplicación preguntará al usuario qué actividad va a realizar a continuación. Dicho formulario se basará en la simple pulsación de un botón dentro de un listado de estos, con la etiqueta asociada a la actividad. Además, se podrá incluir *emojis* al principio de la etiqueta para facilitar la asociación de las actividades, o para su reconocimiento en usuarios con problemas de vista y/o lectura.

3. Al seleccionar la actividad, la aplicación preguntará el estado anímico que el usuario asocia a la actividad de forma idéntica a la anterior: mediante botones con etiquetas y *emojis* que permita asociar las emociones a un pictograma, de forma que sea muy intuitivo el seleccionarla.

4. Una vez seleccionado el estado emocional, la aplicación se cerrará automáticamente y devolverá al usuario al estado en el que se encontraba justo antes de lanzar la aplicación (normalmente el escritorio o cajón de aplicaciones). De esta forma, el usuario podrá seguir su uso normal del dispositivo, mientras que la aplicación guardará en base de datos la información introducida.

   Cabe recalcar que antes del cierre de la aplicación, aparecerá un mensaje modal informando que la sesión ha sido guardada correctamente, para evitar que el cierre sea de forma abrupta, y lleve al usuario a pensar que ha habido algún tipo de error.

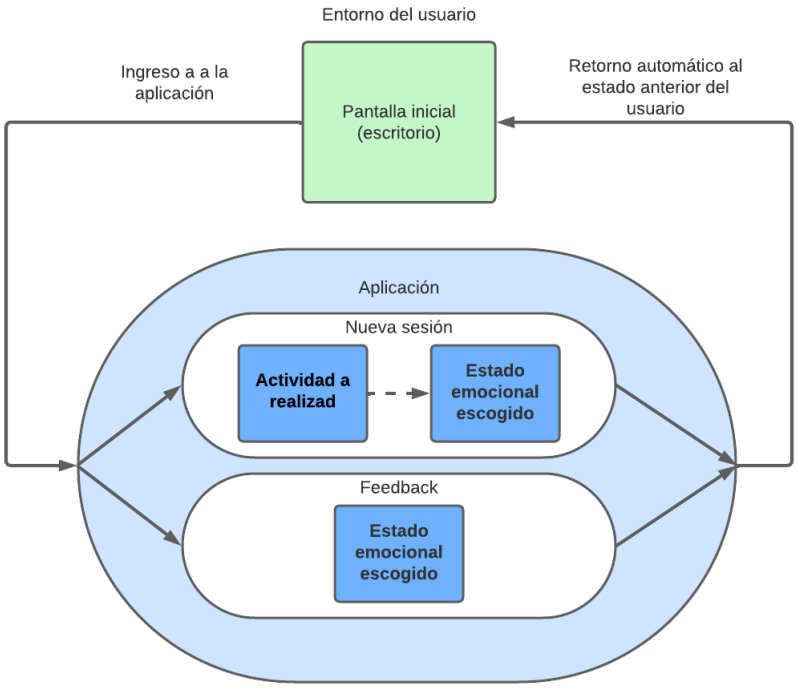

Figura 4.3: Flujo de uso de la aplicación

Dicho flujo responde al caso donde se genere una nueva sesión y haya comenzado una actividad. Para cerrar una, de la forma que hemos explicado en el apartado de *feedback* (4.2.2), el esquema será el siguiente:

1. El usuario accede a la aplicación de la forma que el prefiera (Acceso directo desde el escritorio, relanzar desde las aplicaciones en segundo plano, etc.) tras finalizar la actividad que estaba realizando.

2. La aplicación, al detectar que existe una sesión en curso, muestra el listado de emociones como en el caso anterior. En este caso, preguntado acerca de cual asociaría **tras finalizar** la actividad.

3. Una vez seleccionado el estado emocional, la aplicación se cerrará automáticamente y devolverá al usuario al estado en el que se encontraba justo antes de lanzar la aplicación, normalmente el escritorio o cajón de aplicaciones. De esta forma, el usuario

Figura 4.4: Interfaz de usuario al crear una nueva sesión

podrá seguir su uso normal del dispositivo, mientras que la aplicación guardará en base de datos la información introducida.

4.2.4. Diseño base de datos

Para la persistencia de datos, optaremos por emplear el sistema de gestión de base de datos **SQLite**. Dicho sistema permite ejecutar todos los comandos y consultas estándares de **SQL**, en un formato compacto, reducido y con bajo consumo de recursos. Para nuestro caso, este modelo es más que suficiente.

En la figura 4.6 podemos observar el esquema lógico-relacional con las tablas que componen la base de datos, sus atributos y las relaciones entre ellas. A continuación, detallaremos cada tabla y su uso en la aplicación:

1. **usuario**: Almacenaremos la información relevante en cuanto al usuario (nombre, edad, género, y medio de contacto). Contará con un identificador único que nos permitirá distinguirlos en la aplicación de escritorio. Toda información relevante con el usuario, se encontrará relacionado con esta mediante el *id*

2. **relacion**: Unirá al usuario con los contactos asociados a este en caso de requerir avisar a personas ajenas. Las relaciones dependerán del parentesco con el usuario (hijo, hermano, tíos, etc.) o de si tratamos con profesionales de la salud (cuidador, psicólogo, médico de cabecera, etc.)

Figura 4.5: Interfaz de usuario al cerrar una sesión (*feedback*)

3. **contacto**: Información necesaria para enviar el aviso necesario en caso de detectar alguna circunstancia, con respecto a la información asociada en la tabla.
4. **estadisticas**: Conjunto de valores obtenidos al analizar los vectores registrados en las tablas *AV* y *HR* asociados con el usuario.
5. **contexto**: Tabla con el listado de actividades a elegir por parte del usuario.
6. **mood**: tabla con el listado de emociones a asociar con la actividad seleccionada (*context*). Lleva asociado un campo *icon* que permite guardar un *emoji* a modo de representación visual de la emoción.
7. **sesion**: Tabla principal en la aplicación. Cada uso de esta generará una nueva entrada en la tabla, almacenando las referencias del *context* y *mood* seleccionados por el usuario. Contará con un campo extra para el *mood* donde almacenaremos las sensaciones obtenidas tras finalizar la actividad. Además, guardará la fecha y hora en la comenzó y finalizó la actividad.
8. **evento**: En caso de detectar una situación fuera del patrón normal del usuario (tristeza prolongada en el tiempo), se llevará a cabo cierta/s acción/es. Un caso es el de comunicar a familiares almacenados en la tabla *contacto* según la prioridad establecida en la tabla *relación*. Se almacenará la acción asociada a la sesión donde se disparó, así como los posibles comentarios respecto a la situación.
9. **AV**: Cada entrada en esta tabla almacenará el par arousal/valence registrado durante el uso de la aplicación así como su fecha de registro y la sesión en la cual se obtuvo.

Figura 4.6: Diagrama base de datos

10. **HR**: Esta tabla tendrá el objetivo de almacenar y asociar los registros del ritmo cardíaco con respecto al periodo temporal y la actividad en curso. Dicha tabla corresponde a una futura ampliación a desarrollar por otro compañero (véase el apartado 6.1).

4.2.5. Reconocimiento facial

Durante el uso que se haga de la aplicación, emplearemos la cámara frontal del dispositivo para poder obtener más información relevante del estado en el que se encuentra el usuario. Sobre esta cuestión existen varios trabajos en la materia que permiten comprender la forma en la que podemos pasar de imágenes a una representación numérica representativa, pero en nuestro trabajo no vamos a reimplementar la lógica necesaria por la sobrecarga de trabajo, así como alejarse del objetivo principal de este. En cambio, vamos a recurrir a servicios de terceros que automatizan todo el proceso, y nos otorguen los datos directamente para su tratamiento. Escogemos la herramienta de **Morphcast** (explicada con más detalle en el apartado 2.3.2). Para emplear dicha herramienta, es necesario

generar una clave de licencia (clave API) que permita emplear los servicios. Dicha clave otorga al programador un uso de hasta 2000 minutos mensuales sin ningún tipo de cargo.

En cuanto a la forma de integrarla en la aplicación existente, tiene ciertas características que necesitan ser aclaradas. La aplicación esta concebida entorno a la programación web y su desarrollo se aplica en los entornos HTML/Javascript. Toda lógica a implementar se define mediante código JS embebido dentro de HTML, aunque existen ciertos *boilerplates* que permiten su integración en *fraemworks* como React, Angular y Vue. En el caso de querer integrar dicho servicio a una aplicación web convencional, el programador no tendrá mayor dificultad que la escritura de un par de decenas de lineas nuevas. En nuestro caso (aplicación móvil), deberemos de darle un par de vueltas.

Si leemos la documentación del SDK, obtenemos una escueta guía sobre cómo poder implementarlo en aplicaciones nativas tanto en Android como en iOS. Dichos métodos pasan por implementar funciones que traten de forma explícita la captación de los diferentes *frames* de la cámara, su codificación y envío al SDK, y viceversa (el tratamiento de la recepción de los datos ya procesados). Esta aproximación presenta dos dificultades para nosotros:

1. Implementación tediosa: La codificación de la lógica necesaria para la solución anteriormente explicada necesita de un estudio previo e intensivo sobre cómo funciona internamente la API de dispositivos en Android (micrófono, giroscopio, altavoces y en este caso, **cámara**), así como el formato de las imágenes captadas, su codificación a base64 y su correcto envío al SDK. Todo ello lleva a un consumo de tiempo dentro del trabajo innecesario en comparación a la solución ideada que explicaremos a continuación.

2. Incompatibilidad con la plataforma de desarrollo: Nuestro caso se desarrolla dentro de React Native. Dicho entorno destaca por no requerir crear código nativo de ningún tipo para su funcionamiento, y basarse en el uso de librerías públicas[2] para expandir las funcionalidades. Es obvio que para este caso no disponemos de una librería ya creada que permita usar el SDK dento de React Native, así que su integración pasaría por crear nuevos módulos nativos e importarlo dentro del entorno de trabajo. Esta opción presenta la desventaja de tener que migrar toda la lógica expuesta en la documentación oficial, hacia el entorno de React, lo que aumenta considerablemente la dificultad. Además, tener que optar a esto va en contra de la filosofía adoptada a la hora de trabajar dentro de un *fraemwork* y no directamente desde la plataforma nativa.

Dicho esto, a modo de solución se probó a integrar la funcionalidad de Morphcast como si de una aplicación web se tratara, y embeberla dentro del código React. De esta forma, solo deberemos crear un archivo muy reducido de código HTML/JS (no hace falta incorporar ningún tipo de contenido, esta pseudo página no se mostrará), cargarlo como si de una página normal se tratara y emplear los datos obtenidos ya dentro del entorno React. La forma más fácil de implementarlo consiste en generar el archivo HTML (registrando los eventos de arousal/valence y devolviéndolos de forma que podamos tratarlos), lanzarlo a un pequeño servidor y cargarlo desde React Native mediante un WebView, el cual no es más que una pequeña instancia de navegador encargado de cargar la página web y mostrarla de forma amigable dentro de la aplicación. Si ocultamos el Webview en todo momento que se use la aplicación, obtendremos la funcionalidad que buscamos de forma idéntica a la aproximación nativa. El hecho de tener que cargar el archivo HTML desde un medio ajeno y no de forma local radica en la naturaleza del propio SDK. Este requiere la carga de aplicaciones mediante conexiones SSH para garantizar los

[2]Repositorio más popular: https://www.npmjs.com/

criterios de seguridad requeridos, además de que los navegadores modernos requieren de esta configuración para poder emplear diferentes características técnicas empleadas en el SDK, como puede ser enlaces relativos, *ajax and cors*, almacenamiento local, *service workers*, etc. [24])

El desarrollo e implementación de esta solución se llevó a cabo de forma exitosa, pudiéndose ocultar al usuario mientras se obtiene en un segundo plano un flujo de datos constante. Cabe recalcar que el reconocimiento facial está limitado a un periodo de 30 segundos desde el momento que el usuario accede a la aplicación, a modo de evitar un consumo excesivo en el uso del SDK (o en caso de que el usuario se olvide de desconectar el dispositivo tras su uso), con el consiguiente coste económico.

Para mejorar el producto, buscamos la forma de poder incluir el archivo HTML dentro de los archivos de la aplicación. De esta forma, se desarrollaría íntegramente dentro del dispositivo sin requerir de agentes externos. Para ello, buscamos las formas de crear un **servidor local** durante el tiempo de ejecución de la aplicación y configurarlo de forma que este solo tenga que devolver el archivo precargado. La labor de implementación resulto tediosa debido al hecho de que las librerías ofrecidas para generar el servidor local se encontraban desatendidas, con serias brechas de seguridad y no realizaban su función correctamente, lo cual forzó a crear los módulos nativos necesarios en lenguaje Java (rompiendo el principio por el cual escogimos React, como explicamos anteriormente) y su integración en el resto del código. Si bien es cierto que se logró levantar dicho servidor durante el tiempo de ejecución, y obtener el flujo de datos por parte del SDK, al pasar de un entorno de desarrollo a producción (i.e. versión final) la aplicación perdió dicha funcionalidad. No podemos garantizar exactamente el fallo que ocasionaba esto, pero tenemos la firme convicción de que se debe al hecho de no poder generar un servidor local mediante el protocolo SSL/HTTPS, con el consiguiente rechazo por parte del SDK.

Dadas las dificultades mencionadas, se optó por externalizar la función de servidor mediante los servicios de *hosting* u hospedaje web que permite alojar páginas estáticas en la nube. Existen muchas opciones dentro de estos servicios, siendo el host de Amazon AWS nuestra elección final. Estos *host* no implican coste económico por su uso. De esta forma, basta con conectar el *WebView* al enlace de nuestro *host* personal.

Con todo lo descrito, obtenemos un funcionamiento interno como se muestra en la figura 4.7

4.3 Diseño del back-end

La información almacenada en el dispositivo móvil no tiene forma de ser mostrada ni analizada desde el mismo, dado que este no es el objetivo, y hacerlo desde este, resultaría muy incómodo y poco conveniente. Además, existen ciertos escenarios en la que una persona responsable tenga que llevar el cuidado y atención de varios pacientes. Dar un lugar centralizado para poder analizar, generar informes y llevar un control del trabajo es prácticamente imperativo.

Para crear este *hub* donde almacenar los diferentes registros, vamos a basarnos en las plataformas de escritorio (véase Windows, macOS, o GNU/Linux) las cuales permiten un trabajo más cómodo, y se encuentra más estandarizado su uso en entornos profesionales. En esta primera versión, bastará con que la aplicación sea capaz de almacenar e indexar los usuarios y sus registros generados en la aplicación móvil. Con estos registros, el responsable podrá visualizar los datos relevantes de forma gráfica y de fácil interpretación.

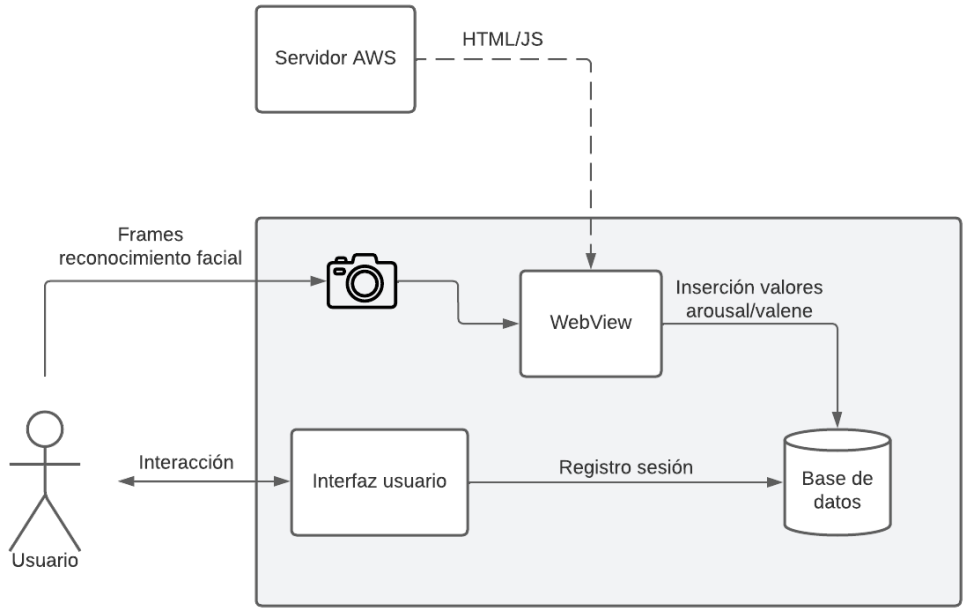

Figura 4.7: Esquema funcionamiento de la aplicación

4.3.1. Volcado de datos

Para almacenar en el *back-end* los datos de los usuarios, hemos tratado de ofrecer un método lo más sencillo e intuitivo posible. Para ello, emplearemos la generación de imágenes QR y el uso de la cámara del teléfono para la comunicación entre ambos dispositivos. Dicho método resulta de lo más sencillo, pues su uso consiste únicamente en apuntar la cámara hacia el código generado por la aplicación de escritorio. Ambas aplicaciones se encargarán de comunicarse entre si como explicaremos a continuación. Para su realización, el usuario presionará el botón "QR", el cual mostrará un mensaje modal con la imagen a escanear (Figura 4.8). Ahora, desde el teléfono, el usuario entrará en el menú de "copia de datos" y, con el marco de cámara mostrado, lo alineará con la imagen anterior. Automáticamente, tras detectarlo, comenzará el envió de datos, tras lo cual se informará al usuario de la finalización de este (Figura 4.9). Una vez hecho esto, en la aplicación de escritorio se actualizará el listado de usuarios (si este fuera uno nuevo), y aparecerá en el desplegable junto al resto.

4.3.2. Informes generados

Mapa arousal/valence

Al seleccionar un usuario en el listado de la parte superior, la aplicación generará de forma automática las gráficas necesarias, y actualizará la pantalla y su subcategorías con la nueva información. Esto se ve más claramente al ir cambiando el usuario seleccionado.

Las diferentes categorías que podemos analizar se agrupan en un *Tabview*, donde cada pestaña seleccionada cambia las gráficas mostradas. Mostrarlas todas a la vez y en la misma pantalla podría llegar a ser confuso y poco amigable visualmente hablando, por lo cual hemos optado por esta forma de representación donde en cada momento solo una gráfica ocupa toda el área de la aplicación.

La primera de ellas, y la que se selecciona de forma predeterminada, es la gráfica bidimensional de los vectores **arousal y valence** (4.10).

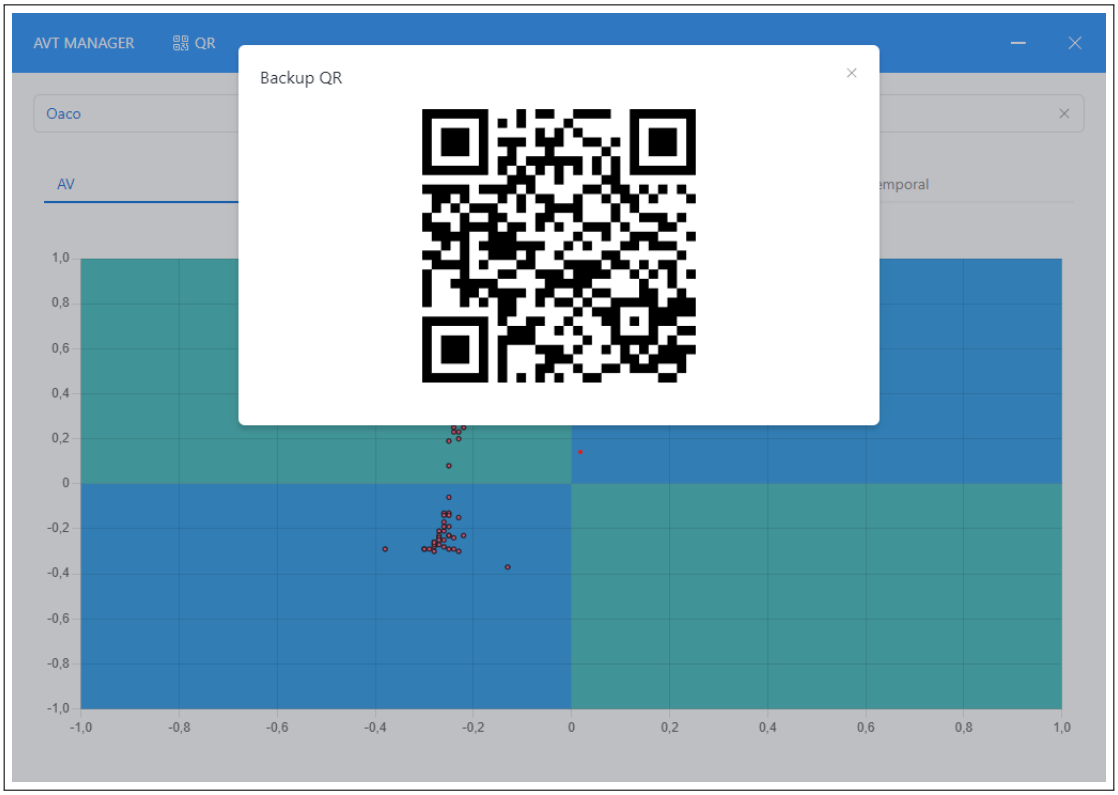

Figura 4.8: Captura de AVT Manager, código QR generado

Esta gráfica tendrá la función de mostrar todas las parejas de valores recabados por la API de MorphCast mediante la representación por cuadrantes. La tabla donde mostrar se formará por los ejes X e Y acotados en los rangos [-1, 1], máximos que pueden tomar cada componente. Los diferentes cuadrantes representarán los estados valencia positiva/negativa - activación positiva/negativa (combinación de 2^2 posibilidades). De esta forma, se puede obtener de un primer vistazo una estimación de la zona más frecuente en la que se encuentra el usuario, y visualizar situaciones anómalas (una pequeña nube de puntos en cierta área; puntos sueltos a lo largo de la tabla es normal debido a lecturas erróneas).

Actividades/Feedback

Al seleccionar la segunda pestaña, obtendremos un histograma donde se mostrarán las emociones asociadas a cada actividad disponible (4.11). El conteo de emociones se agrupará a modo de pila, de forma que se pueda reconocer de un vistazo las preferencias del usuario respecto a las actividades.

La tercera pestaña muestra exactamente la misma gráfica, variando los datos de entrada; en este caso leeremos los registros de retroalimentación del usuario al finalizar la actividad (4.12). Mantener estas dos pestañas de igual forma permite una comparativa mucho más fácil sobre la evolución del usuario antes y después de realizar la actividad (las gráficas muestran una discreta animación, evitando brusquedad en el cambio de datos y ayudando a observar el cambio de las gráficas).

Evolución temporal arousal/valence

Para finalizar, al seleccionar la última pestaña, se nos mostrará una gráfica temporal, donde podremos observar el promedio de las variables arousal y valence para cada día,

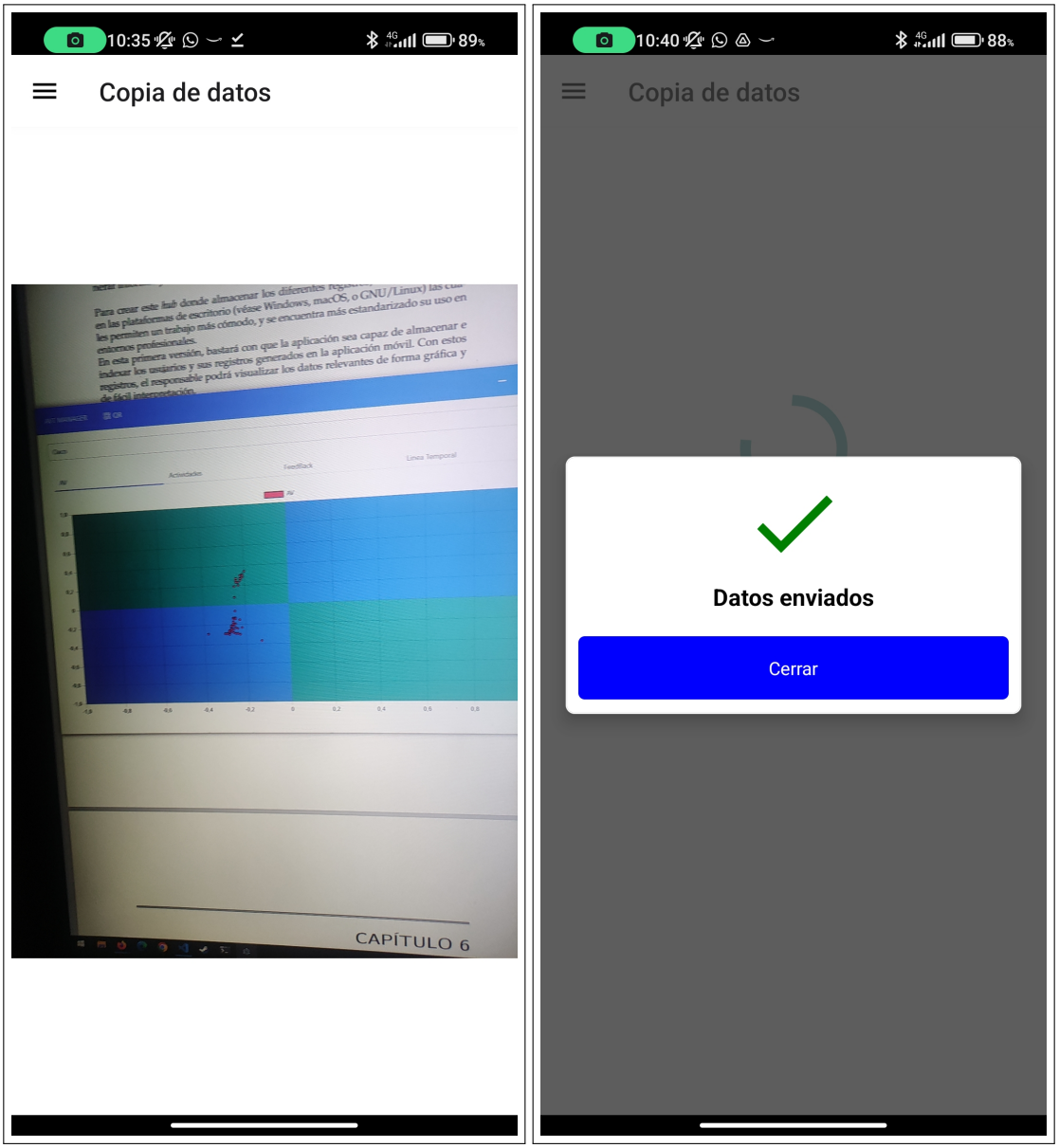

Figura 4.9: Captura de AVT, pantalla de *backup*

así como su evolución a lo largo del uso de la aplicación (4.13). La principal función de esta gráfica es la de poder observar cambios en el patrón de las dos variables, asociarla a un periodo temporal, y poder discernir las causas de este cambio (esta última labor del responsable).

Figura 4.10: Captura de AVT Manager, tabla AV

Figura 4.11: Captura de AVT Manager, tabla actividades

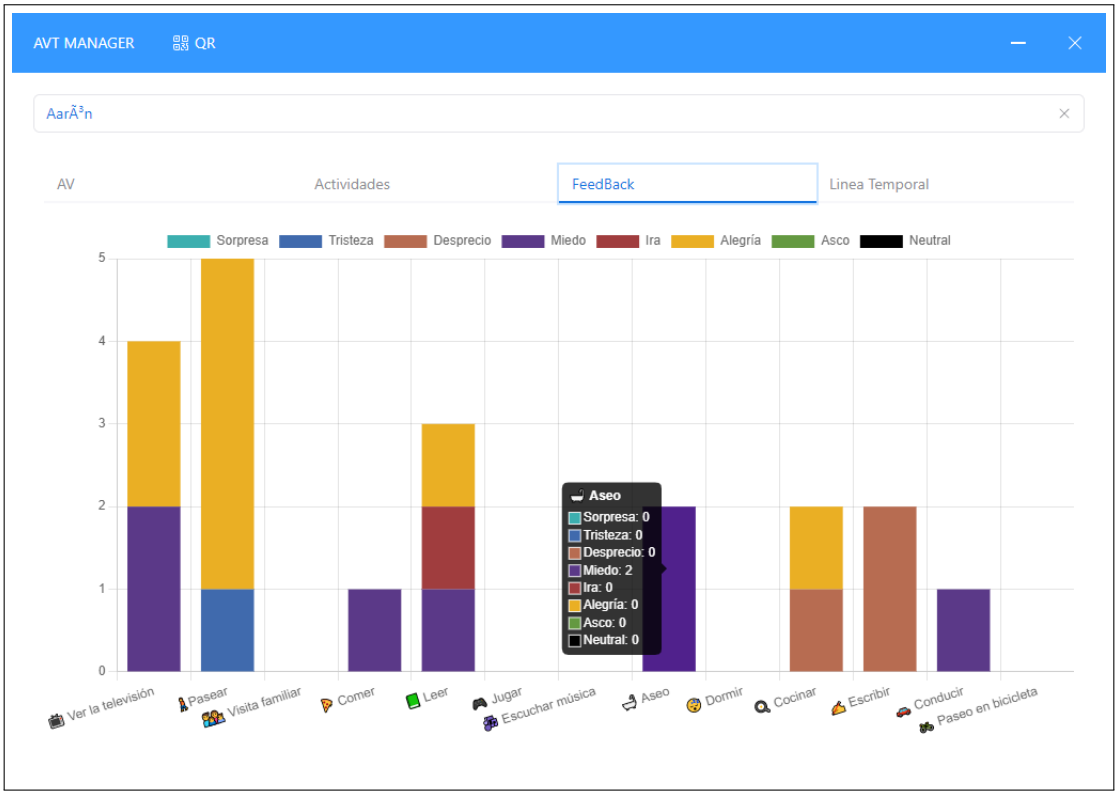

Figura 4.12: Captura de AVT Manager, tabla feedback

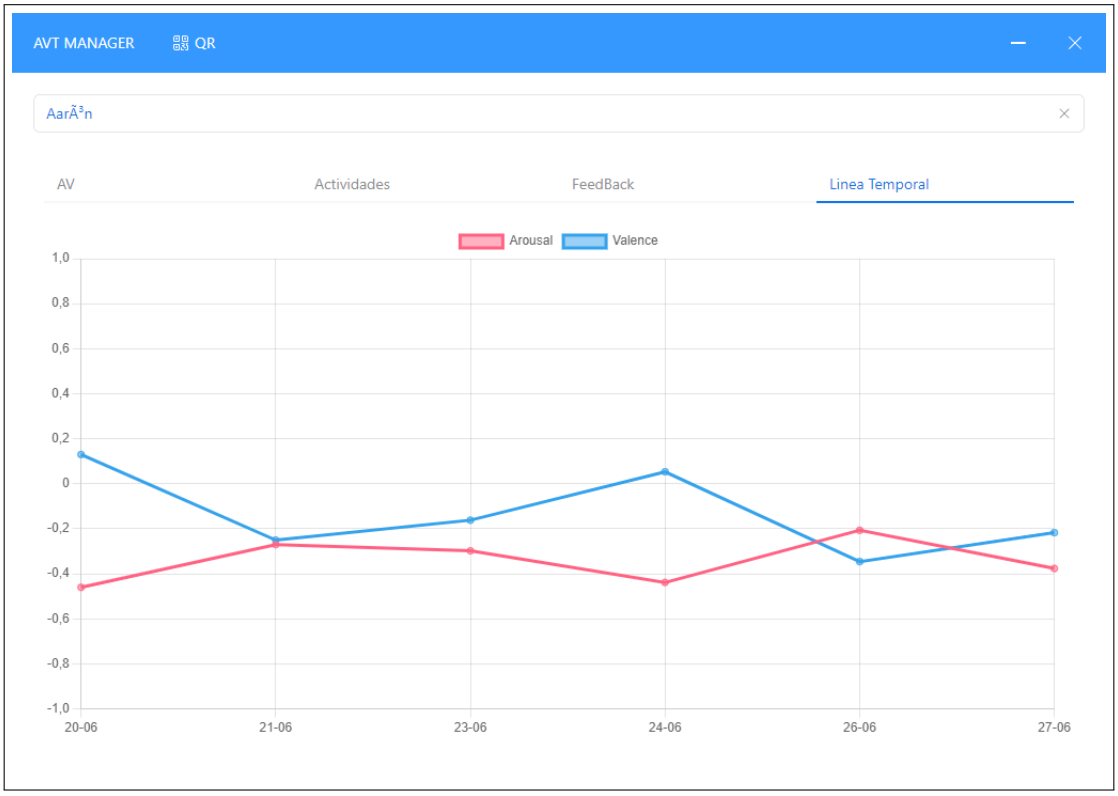

Figura 4.13: Captura de AVT Manager, tabla temporal arousal y valence

CAPÍTULO 5
Desarrollo e implementación

Para desarrollar nuestra aplicación siguiendo los criterios establecidos en el anterior capítulo, necesitaremos configurar nuestro entorno de desarrollo.

5.1 Entorno de trabajo

Crear aplicaciones mediante el uso de **React Native** requiere de las mismas plataformas de desarrollo que si optásemos por la metodología nativa. Aunque generar aplicaciones con React evita escribir código Java/Kotlin y manipular funcionalidades a bajo nivel, su compilación requiere obviamente de todas estas herramientas para poder realizar las pruebas pertinentes en dispositivos reales o emuladores). Para ello, encontramos en la página de React Native los pasos necesario para poder instalar todas las dependencias necesarias par el desarrollo. A modo de resumen, se requiere instalar:

1. **Node.JS y JDK**[1]: Requeridos tanto para el *binding* de JS (generación de código óptimo para React Native, puesto que JS es un lenguaje interpretado) como la compilación del código Java. Para este último, se puede optar por **OpenJDK** si se prefiere las soluciones basadas en *open source*.
2. **Android Studio**[2]: IDE oficial para el desarrollo de Android. Si bien no emplearemos dicho entorno para desarrollar nuestra aplicación, React Native si que requiere de los diferentes SDK que ofrece. En especial, el SDK de Android 33 y las APIs de Google para la compilación en plataformas x86. Además, Studio ofrece la posibilidad de instalar y configurar un emulador de dispositivos Android, con la capacidad de emplear aceleración por *hardware* (Intel HAXM) donde realizar las pruebas necesarias de nuestra aplicación.
3. **Watchman**[3]: Watchman permite al entorno de desarrollo detectar cambios en el sistema de archivos en tiempo real. De esta forma, podremos realizar cambios en el código y verse reflejados al instante en nuestra aplicación mientras esta esté ejecutándose (emulador o dispositivo real). Esto se debe gracias a la naturaleza de JS de interpretar el código. Cambios en los módulos nativos requerirán relanzar la aplicación.

A partir de aquí, solo resta al programador escoger su editor de textos favorito para comenzar a desarrollar. En nuestro caso, hemos optado por **Visual Studio Code**[4]. Code

[1] Página oficial: https://www.oracle.com/es/java/
[2] Página oficial: https://developer.android.com/studio
[3] Página oficial: https://facebook.github.io/watchman/
[4] Disponible en: https://code.visualstudio.com/

(de ahora en adelante) nos ofrece un editor modular, donde podremos ajustar nuestra experiencia con diferentes extensiones para disponer de las funcionalidades necesarias. En nuestro caso, empleamos las extensiones de React Native que nos ofrecen la funcionalidad de *intellisense* (Detección y autocompletado de palabras) relacionado con el código React, así como diferentes *snippets* (fragmentos de código reutilizables). También hemos optado por emplear extensiones de gestión de base de datos **SQLite**, las cuales permiten leer las tablas presentes en archivos ".db" y realizar diferentes *queries* sobre este. Además, Code ofrece un entorno adaptable al usuario, con la posibilidad de editar los temas, colores y fuentes empleados. Esto que puede parecer algo superfluo, resulta vital cuanto el programador va a pasar largas jornadas delante de la pantalla. La fatiga visual supone un serio problema en el oficio. También integra el uso de varias terminales simultaneas dentro de la interfaz, cosa que resulta muy conveniente a la hora de desarrollar, pues se hace un uso intensivo de estas (lanzar al dispositivo la aplicación, generar el *binding* de React/JS, etc.).

Como curiosidad, recalcar que Code está desarrollado por la empresa de renombre Microsoft, e implementada íntegramente en Electron.JS, *fraemwork* que empleamos nosotros en la creación de la aplicación de gestión. Esto muestra la madurez y confianza que existen en estos entornos, siendo una apuesta segura para la creación de aplicaciones.

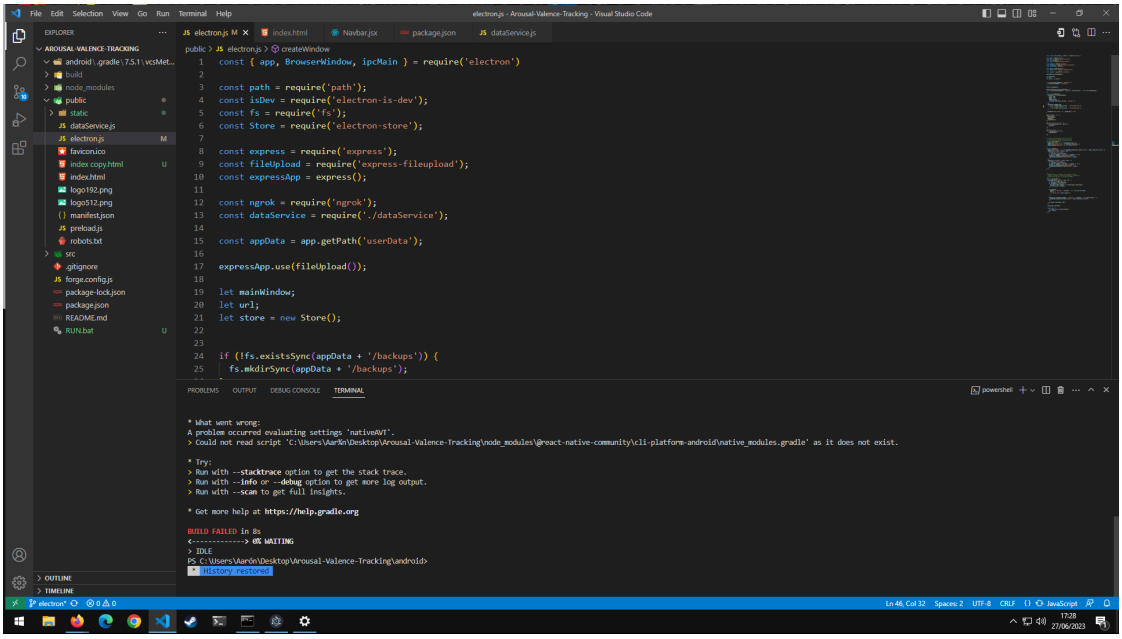

Figura 5.1: Instancia de Visual Studio Code

5.2 Arquitectura de React Native

Para poder desarrollar en React, primero deberemos entender como se atienden las peticiones que podamos realizar al sistema. A groso modo, en este contexto no requerimos codificar ningún tipo de código que interactúe directamente con el dispositivo. Esto es, no necesitamos comunicarnos con las APIs de los diferentes periféricos, pues es esta la tarea de React; ofrecer todas las funcionalidades en un entorno Javascript. La forma en la que realiza esto esa mediante el uso del *bridge*, que actúa a modo de "intérprete" entre el contexto Javascript y el contexto nativo (véase la figura 5.2).

Debemos entender que el funcionamiento de este tipo de aplicaciones pasa siempre por la ejecución de código Java/Kotlin, y el *bridge* se encarga de traducir nuestras peticio-

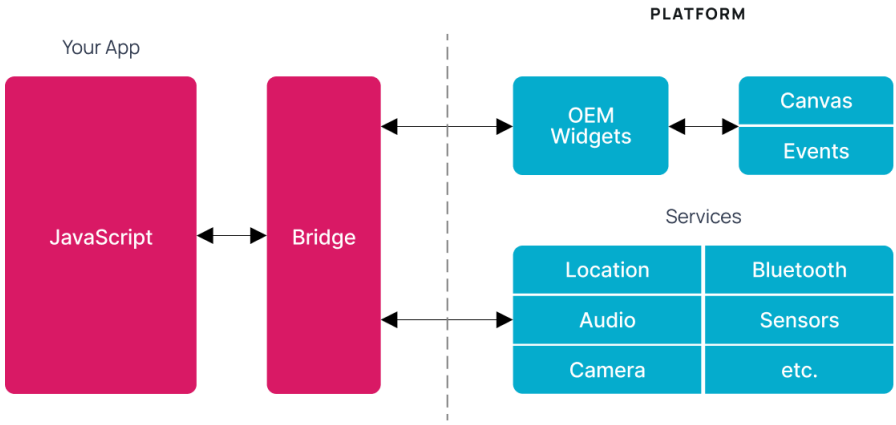

Figura 5.2: Diagrama arquitectura React Native

nes hechas desde Javascript hacia el contexto nativo mediante las llamadas a los módulos pertinentes. Si indagamos en el código generado por React, vemos que su base es la dependencia de un listado de módulos predefinidos en Java, junto a la instancia de una única *activity* (nombre dado a una pantalla de la interfaz de usuario).

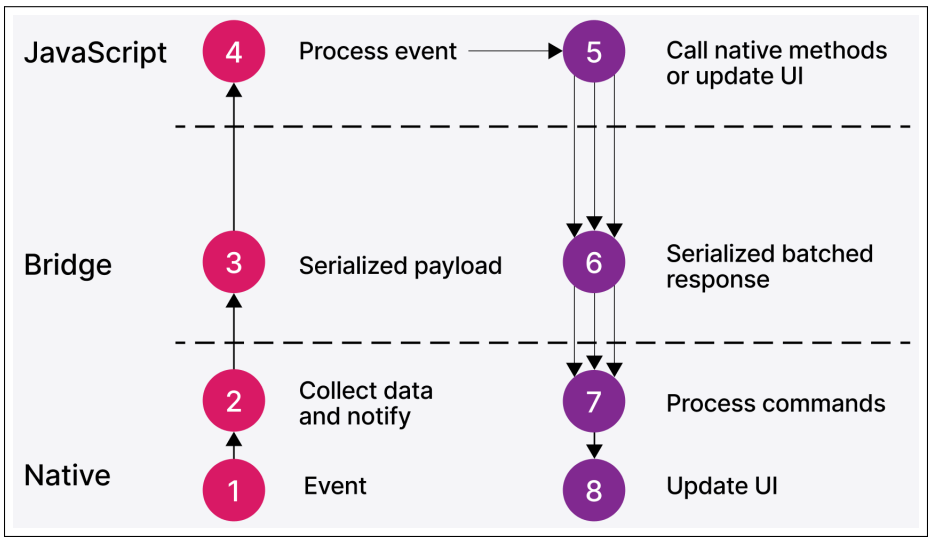

Figura 5.3: Diagrama lógica de negocio React Native

En cuanto a la comunicación, React emplea la técnica de **intercambio de mensajes serializados**. De esta forma, se obtiene una comunicación asíncrona que evita bloqueos al procesar las peticiones, y mantiene los hilos de React y de código nativo separados. Para ser más exactos, existen **tres** hilos de ejecución: el correspondiente a React Native (Javascript), el Bridge y el hilo de código nativo. El esquema 5.3 muestra de forma gráfica los pasos de comunicación entre hilos.

5.3 Modo de trabajo

Boilerplate

Una vez aclarado las anteriores cuestiones, podemos comenzar con el desarrollo puro de la aplicación. Para esto, podemos evitar la creación manual de los archivos base mediante el uso del comando

npx react-native@latest init <Nombre-proyecto>

Dicho comando creará un directorio con el nombre del proyecto, y generará los archivos básicos para lanzar una aplicación vacía sobre la cual trabajar (*boilerplate*)

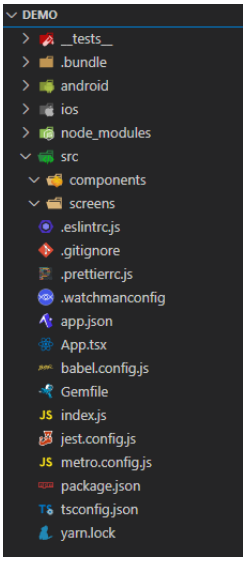

Figura 5.4: Diagrama lógica de negocio React Native

Los elementos del directorio que serán relevantes para nosotros son:

1. **components**\: crearemos aquí todos los componentes que empleemos en las pantallas a mostrar al usuario. Un componente representa elementos reutilizables en diferentes contextos. Por ejemplo, se puede reutilizar un componente que muestre una lista de bonotes con diferentes datos de entrada.

2. **screens**\: Definiremos las diferentes pantallas sobre la que el usuario pueda navegar, como una pantalla de registro, el menú principal, ajustes, etc. Por normas de estilo, se crearán subdirectorios para cada una de estas pantallas.

3. **package.json**: Será el guión sobre la que crearemos nuestra aplicación. En el archivo definiremos metadatos, como nombre, versión, autor, repositorio, además de definir las dependencias a instalar tanto en desarrollo como en producción. Es vital mantener este archivo correctamente, pues la instalación de paquetes pasa por este archivo, y una mala configuración puede imposibilitar el desarrollo (conflictos de paquetes, versiones de librerías no soportadas, etc.).

Con estos tres elementos, podemos comenzar nuestro desarrollo. Profundizar en el resto del directorio es una excelente manera de entender el funcionamiento de la aplicación, pero para la memoria nos centraremos en estos.

JSX

La creación de un componente en React Native pasa por el lenguaje JSX, el cual podemos definir de forma básica como “Javascript con HTML integrado”. El objetivo final de cada uno de estos componentes es devolver una estructura con estilo HTML que representará aquello que el usuario visualizará.

```
import { View, Text, StyleSheet } from 'react-native'

function Header({ title }) {
return (
    <View style={styles.header}>
        <Text style={styles.text}>
            {title}
        </Text>
    </View>
);
}

const styles = StyleSheet.create({
    header: {
        alignItems: 'center',
        paddingVertical: 40
    },
    text: {
        color: 'black',
        fontSize: 25,
        fontWeight: 'bold'
    }
})

export default Header;
```

En el código anterior de ejemplo, la función **Header** se encargará de devolvernos el elemento a renderizar por el componente que lo llame. La estructura de etiquetas define los elementos que lo integran, así como los contenedores que lo componen. En este caso, se retorna un texto dentro de un contenedor de vista (*view*). Los elementos de la función *return* pueden estar estilizados (ajuste de tamaño, márgenes entre elementos, colores, etc.) mediante el uso de la sintaxis de CSS. Además, React ofrece el objeto **StyleSheet** con el cual podemos definir todos los estilos necesarios y acceder desde la declaración de los elementos en la función **return**, dándonos un código mucho mas legible y amigable.

Por último, es importante exportar correctamente la función con el componente. En archivos más extensos y complejos, podemos contar con multitud de funciones dentro del mismo archivo.

Gestión base de datos

Para la persistencia de datos, emplearemos la librería de **react-native-sqlite-storage** [5], la cual trabaja mediante llamadas asíncronas. Por lo cual, el tratamiento de los datos pasará por las **promesas** de Javascript. Las promesas son los nuevos mecanismos que vienen a sustituir la notación *await, async*. La ejecución de comandos SQL (como consultas, inserción en base de datos, actualizaciones de entradas, etc.) pasa por definir métodos asíncronos que definan el tratamiento de los datos durante su ejecución, así como el tratamiento de posibles excepciones. La mejor forma de definir los comandos es la creación de un único archivo con todos los comandos que podamos emplear durante la aplicación

[5] Disponible en: https://github.com/andpor/react-native-sqlite-storage

```
export async function getContext() {
    const db = await getDBConnection();
    const context = [];

    await db.transaction((txn) => {
        txn.executeSql(
        'SELECT * from contexto', [],
        (tx, result) => {
            for (let i = 0; i < result.rows.length; i++) {
                context.push(result.rows.item(i))
            }
        },
        (err) => console.log(err));
    })
    return context;
};
```

En el anterior fragmento de código podemos observar la forma de trabajo anteriormente descrita. La llamada a esta función (getContext()) devolverá un objeto promesa. En el fragmento de código donde se llame, se tendrá que definir el tratamiento que se hará de los datos a devolver (mediante *callback*, o función de orden superior). Siguiendo el ejemplo de *getContext()*, uno de los usos dados en la aplicación es el siguiente

```
getContext().then((contextList) => {
    var tmp = [];
    for (let i = 0; i < contextList.length; i++) {
        tmp.push({
            id: contextList[i]["id_contexto"],
            title: contextList[i]["name"]
        })
    }
    setDataList(tmp);
});
```

5.3.1. AVT Manager

Todo lo descrito hasta el momento se puede aplicar al desarrollo de nuestra aplicación de escritorio para la gestión de los perfiles de usuario. Electron.JS emplea los mismos elementos de desarrollo, como el uso de librerías **npm** definidas mediante el archivo package.json y la gestión de base de datos mediante la misma librería **sqlite**[6].

Para la creación de interfaces de usuario, se ha optado por React.JS, la cual sigue el mismo esquema explicado para React Native, por lo cual no es menester detallar la creación de la interfaz escritorio.

[6]Disponible en: https://github.com/TryGhost/node-sqlite3

Express.JS

Un detalle que si merece tener en cuenta es la comunicación entre dispositivo móvil y ordenador. Entre todas las interfaces disponibles, optamos por la comunicación mediante el envío de datos por protocolo HTTP. Dicha aproximación permite que el envió de copia de datos entre ambos dispositivos no requiera de ningún tipo de configuración, ni pertenecer a una misma red local. Basta con que ambos tengan acceso a internet. Para ello, deberemos crear un *endpoint* de tipo POST en nuestra aplicación de escritorio que permita almacenar la información del usuario en disco. Express.JS [7] permite la definición de estos métodos con un par de decenas de líneas de código, siendo la lógica necesaria totalmente oculta al programador, dejando a este solo con la tarea del tratamiento de datos.

Al generar nuestro servidor embebido en la aplicación, la dirección por defecto para acceder a este es **http://localhost:«puerto»** debido a que no deja de ser un servidor local a la máquina. Para poder acceder a él, debería de estar el dispositivo móvil dentro de la misma red local. Podemos evitar esto si tunelamos nuestra aplicación hacia un dominio público. Existen librerías encargadas de llevar a cabo esto, como es nuestro caso con **ngrok** [8]. Dicha librería permite redireccionar una aplicación (como puede ser la creada con Express.JS) hacia un dominio público que ofrece. La desventaja de esto es que el dominio se genera mediante un *string* aleatorio cada vez que se lanza la aplicación. Pero en nuestro caso, esto resulta ser una ventaja. El uso del código QR radica en este fenómeno: empleamos la imagen para transmitir del ordenador al móvil la dirección URL de una forma automática y sencilla, al requerir comunicarla en cada uso de AVT Manager. La gran ventaja de esto es la de no disponer de una dirección estática, si no aleatoria en cada uso, evitando posibles ataques que comprometan la seguridad de los datos. Además, dicha dirección solo estará publica durante el ciclo de uso de la aplicación.

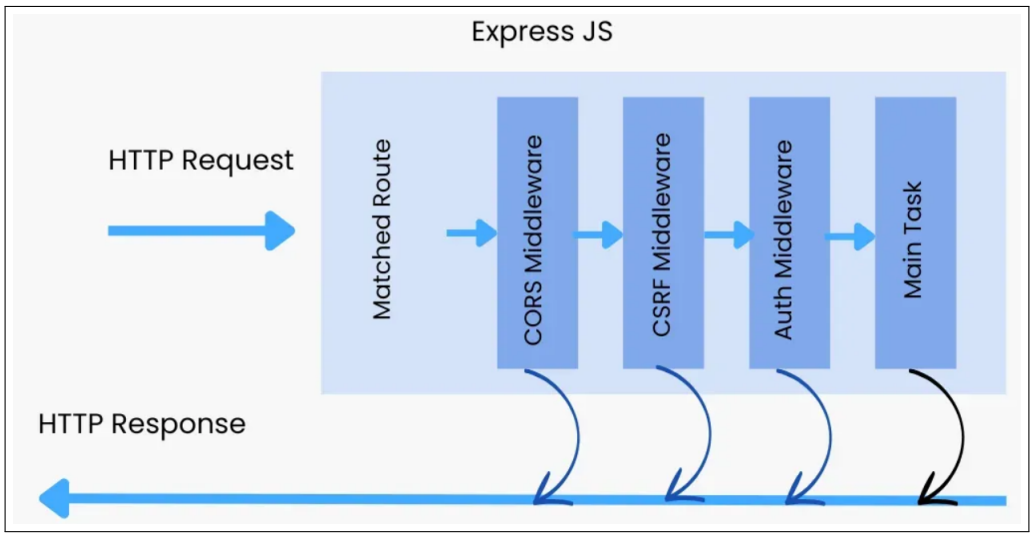

Figura 5.5: Diagrama Express. Fuente [11]

[7] Disponible en: https://github.com/expressjs/express
[8] Disponible en: https://github.com/bubenshchykov/ngrok

CAPÍTULO 6
Conclusiones y futuras ampliaciones

En este trabajo hemos desarrollado una aplicación operativa que cumple todos los objetivos planteados inicialmente. Se trata de una aplicación con una interface muy simple, que consume muy pocos recursos, capaz de funcionar con versiones de android de hace más de 7 años y muy poco intrusiva.

La serie de datos temporales se almacena correctamente en el dispositivo móvil y se transfieren con mucha rapidez y sencillez al back-end. En las pruebas realizadas, se ha comprobado la corrección de las actividades y estados emocionales almacenados en la base de datos y transferidos al back-end para generar informes. La aplicación es fácilmente personalizable para priorizar las actividades que realiza el usuario más frecuentemente. Los datos obtenidos permiten observar zonas de alta concentración en cuanto a registros arousal/valence. Esto permite conocer el estado emocional más habitual en el usuario, y facilita la detección de datos anómalos y su posterior análisis (como determinar qué factor lo genera). Todo el proceso se realiza respetando el RGPD y la LOPDGDD.

La aplicación funciona de forma fluida, con tiempos de respuesta cortos. Esto hace que en apenas unos minutos el usuario aprende a usarla, y la integración en su rutina diaria se hace de forma amena. La integración entre aplicación móvil y escritorio resulta muy sencilla e intuitiva. Las copias de datos no toman más de un par de segundos y la carga y la generación de informes gráficos se realiza de forma instantánea.

En cuanto al proceso de desarrollo de la aplicación, vemos como las soluciones elegidas (*fraemwork*) han supuesto un gran avance especialmente considerando el tiempo y personal disponible para el desarrollo. La agilización del proceso de desarrollo supone un gran impulso en el surgimiento de pequeñas empresas (**pymes** en España), e incluso en los trabajos de carácter individual, como la creación de software libre, o proyectos de tiempo y personal reducido, como el presente **TFG**.

Todo ello hace pensar que existe un gran abanico de posibilidades a explotar dentro de los dispositivos móviles, no solo enfocado en el ocio, entretenimiento y mensajería instantánea, si no también una forma de acercar los cuidados médicos a las personas que lo necesitan (centrándonos en las personas de edades avanzadas en este caso). Los *smarthphones* se han integrado a nuestras vidas de forma total, siendo una extensión más de nosotros mismos. Debemos de dejar de verlos como un mero dispositivo de entretenimiento y explotar todas las ventajas que nos puede ofrecer a lo largo de nuestra vida puesto que, los *smartphones* no van a desaparecer de nuestras vidas. El concepto evolucionará y nuevos dispositivos aparecerán en el mercado, pero la idea de llevar un ordenador en nuestro bolsillo será algo que no veremos desaparecer. Además, disponer de estas herramientas nos permite disolver las barreras entre las personas más abandonadas por la

sociedad. No solo es una cuestión de monitorizar y tratar a las personas en riesgo de soledad, es el hecho de que un acercamiento con una persona, aunque sea por medios digitales, puede suponer un gran estímulo. Estos medios pueden hacer ver a las personas que no están solas, que existe gente detrás de ellos con interés por su bienestar.

En cuanto al autor, el trabajo ha permitido afianzar los conocimientos aprendidos tanto en la carrera como en los diferentes proyectos llevados a cabo. En especial, se han afianzado conocimientos de diferentes materias, con respecto a las plataformas web, conceptos como asincronía, enrutamiento de datos (Tecnología de Sistemas de Información en la Red, **TSR**), la persistencia de datos y su correcta organización y tratamiento (Bases de datos y sistemas de información, **BDI**) y los protocolos de comunicación por red necesarios para enviar información de forma segura (Redes de computadores, **RED**).

6.1 Futuras ampliaciones

El diseño modular de la aplicación permite incorporarle nuevos módulos de reconocimiento de emociones, de comunicación y otras funcionalidades que se desee en el futuro. Las lógicas restricciones temporales asociadas a la realización de un TFG nos obligaron a limitar la cantidad de módulos que se incorporaban a la aplicación. Pero gracias al carácter modular que explicamos en el apartado 3.2, estas funciones pueden ser añadidas de forma sencilla y con una fácil integración entre los demás componentes. Algunas ampliaciones que nos gustaría tener en cuenta de cara al futuro desarrollo de la aplicación son los siguientes

Nuevos dispositivos de reconocimiento Durante nuestro trabajo nos hemos centrado en el reconocimiento facial como fuente de información con respecto al estado emocional del usuario. Sin embargo, existen otra características fisiológicas que podemos observar en un paciente. Un caso interesante es en el monitoreo del ritmo cardíaco. El hecho de que el trabajo se encuentre dentro de las labores de investigación de un departamento de universidad (como detallamos en 1.2) permite que nuevos integrantes en el equipo puedan tomar la aplicación como punto de partida para sus trabajos de investigación. Mientras se desarrollo el trabajo, un alumno de la ETSINF decidió realizar su TFG basándose en el monitoreo del ritmo cardíaco como medio de información respecto al estado afectivo del usuario, por lo cual ya existe en marcha un proyecto de ampliación, cuyo resultado se podrá observar en un corto plazo de tiempo. De esta forma, se establecieron puntos en común para trabajar simultáneamente en la misma aplicación, como la reserva de una tabla dedicada a los registros cardíacos asociadas a la sesión y a los registros faciales correspondientes. Esto amplía la información con respecto al usuario. Por ejemplo, una brusca aceleración del ritmo, puede sugerir la comunicación de una noticia impactante a este, y empleando los registros faciales para determinar si se trata de una situación positiva o negativa para el usuario. Además, un monitoreo constante del corazón puede permitir la predicción de futuras afecciones, e incluso situaciones de riesgo para el usuario, como arritmias (véase la última ECG en la figura 6.1).

Avatar digital y avisos La idea de introducir avatares esta motivada por una interacción más cercana con el usuario, apoyándole en los momentos que se detecte un nivel de tristeza excesivo. Su implementación requerirá del renderizado del personaje virtual, así como dotar al sistema de capacidad de reconocimiento de voz, procesamiento de la información y generación de texto coherente a modo de respuesta al usuario (*text-to-speech*). La mejor solución al problema sería emplear las APIs dispuestas por la empresa **OpenAI**, para interactuar con el modelo de len-

Figura 6.1: Ecocardiogramas de ejemplo. Fuente [12]

guaje **ChatGPT**, que tanto auge ha tenido en los últimos tiempos. De hecho, dentro del equipo de investigación se encuentran varios miembros investigando con dicha herramienta para la interacción con sujetos en peligro de aislamiento social, por lo cual esta es una característica que probablemente se lleve a cabo.

Contextos personalizados El diseño de la aplicación establece que los contextos/actividades que se puedan seleccionar sean cargadas desde la base de datos. La aplicación viene con una BBDD prepoblada con varias etiquetas de actividades, así como las 7 emociones de *Ekman* (más una emoción asociada a un estado neutro). Con este modelo, resultaría muy sencillo de implementar la funcionalidad necesaria para que el usuario o responsable introdujera nuevas etiquetas a la base de datos, y personalizar la experiencia del usuario. La creación de etiquetas implica que los perfiles de usuarios divergieran en cuanto a los registros de actividades, haciendo que su análisis y comparación pudieran resultar poco fiables. Además, la creación de estas etiquetas por parte del usuario puede llevar a que se inserten etiquetas poco concisas, que no permitan al usuario asocial emociones (por ejemplo, la acción "veranear.ªbarca muchas actividades, y asociar una única emoción puede ser difícil). Esta cuestión deberá pasar por la consulta y estudio de los psicólogos asociados al programa, pues son estos quienes tienen la última palabra, y si ellos consideran necesario las etiquetas dinámicas, se implementarán de forma rápida y sencilla.

Bibliografía

[1] "Pirámide de población," visitado el 06/06/23. [Online]. Available: https://datosmacro.expansion.com/diccionario/piramide-de-poblacion

[2] "Pirámides de población de españa: ayer, hoy y mañana," 2022, visitado el 06/06/23. [Online]. Available: https://www.ine.es/infografias/infografia_dia_poblacion.pdf

[3] B. Alfonso, E. Vivancos, and V. Botti, "Toward formal modeling of affective agents in a bdi architecture," *ACM Trans. Internet Technol.*, vol. 17, no. 1, jan 2017. [Online]. Available: https://doi.org/10.1145/3001584

[4] E. Vela, M. Clèries, V. A. Vella, C. Adroher, and A. García-Altés, "Análisis poblacional del gasto en servicios sanitarios en cataluña (España):¿ qué y quién consume más recursos?" *Gaceta Sanitaria*, vol. 33, no. 1, pp. 24–31, 2019.

[5] M. Yarwood, "Psychology of human emotion: An open access textbook," 2022, visitado el 27/06/23. [Online]. Available: https://psu.pb.unizin.org/psych425/chapter/circumplex-models/

[6] E. G. Fernández Abascal, B. García Rodríguez, M. P. Jiménez Sánchez, M. D. Martín Díaz, and F. J. Domínguez Sánchez, *Psicología de la Emoción*. Editorial Centro de Estudios Ramón Areces, ISBN: 978-8480049085.

[7] M. Nowak, "Flutter vs. react native in 2023 — detailed analysis," 2023, visitado el 27/06/23. [Online]. Available: https://www.nomtek.com/blog/flutter-vs-react-native

[8] C. Nutt, "Miitomo: What's nintendo trying to do with its first smartphone app?" 2015, visitado el 13/06/23. [Online]. Available: https://www.gamedeveloper.com/business/miitomo-what-s-nintendo-trying-to-do-with-its-first-smartphone-app-

[9] R. Fernández, "Distribución porcentual de los usuarios de whatsapp en españa por edad," 2023, Visitado el 06/06/23. [Online]. Available: https://es.statista.com/estadisticas/576109/porcentaje-de-los-usuarios-de-whatsapp-en-espana-en-por-edad/

[10] "Smartphones in spain," 2022, visitado el 06/06/23. [Online]. Available: https://www.statista.com/study/31675/smartphones-in-spain-statista-dossier/

[11] Turing, "How to build middleware for node.js: A complete guide," 2022, visitado el 27/06/23. [Online]. Available: https://www.turing.com/kb/building-middleware-for-node-js

[12] L. B. Mitchell, "Introducción a las arritmias," *MD, Libin Cardiovascular Institute of Alberta, University of Calgary*, 2023. [Online]. Available: https://www.msdmanuals.com/es/hogar/trastornos-del-corazÃşn-y-los-vasos-sanguÃ■neos/arritmias/introducciÃşn-a-las-arritmias

[13] D. Cardona Arango and E. Peláez, "Envejecimiento poblacional en el siglo xxi: oportunidades, retos y preocupaciones," *Revista Salud Uninorte*, vol. 28, no. 2, pp. 335–348, 2012.

[14] "¿cuáles son las prioridades de salud de las personas mayores en tiempos de pandemia?" visitado el 24/06/23. [Online]. Available: https://www.isglobal.org/-/-cuales-son-las-prioridades-de-salud-de-las-personas-mayores-en-tiempos-de-pandemia-

[15] A. Pico, J. Taverner, E. Vivancos, A. Raya, V. Botti, and A. Garcia-Fornes, "A monitoring agent for advancing elderly care through mobile health technology," *PAAMS '23, 21st International Conference on Practical Applications of Agents and Multi-Agent systems*, Guimarães (Portugal) | 12th - 14th July, 2023, pendiente de publicación.

[16] MorphCast, "Coca-cola | transform your feelings into art," 2022, visitado el 06/06/23. [Online]. Available: https://www.morphcast.com/showcase/cocacola/

[17] B. Tag, Z. Sarsenbayeva, A. L. Cox, G. Wadley, J. Goncalves, and V. Kostakos, "Emotion trajectories in smartphone use: Towards recognizing emotion regulation in-the-wild," *International Journal of Human-Computer Studies*, vol. 166, p. 102872, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1071581922000982

[18] J. M. De la Serna, *Alexitimia. Un mundo sin emociones*. Tektime, iSBN: 978-8873046547, 2018.

[19] J. PRINZ, "Which emotions are basic?" in *Emotion, Evolution, and Rationality*. Oxford University Press, 04 2004. [Online]. Available: https://doi.org/10.1093/acprof:oso/9780198528975.003.0004

[20] A. Mehrabian, "Pleasure-arousal-dominance: A general framework for describing and measuring individual differences in Temperament," *Current Psychology*, vol. 14, no. 4, pp. 261–292, Dec. 1996. [Online]. Available: https://doi.org/10.1007/BF02686918

[21] P. J. Lang, M. M. Bradley, and B. N. Cuthbert, "International affective picture system (IAPS): Affective ratings of pictures and instruction manual," *Technical Report A-6. University of Florida*.

[22] "Reglamento (ue) 2016/679 del parlamento europeo y del consejo de 27 de abril de 2016 relativo a la protección de las personas físicas en lo que respecta al tratamiento de datos personales y a la libre circulación de estos datos y por el que se deroga la directiva 95/46/ce (reglamento general de protección de datos)," 2016. [Online]. Available: https://eur-lex.europa.eu/legal-content/ES/TXT/?qid=1532348683434&uri=CELEX%3A02016R0679-20160504

[23] "Ley orgánica 3/2018, de 5 de diciembre, de protección de datos personales y garantía de los derechos digitales," «BOE» núm. 294, de 06/12/2018. [Online]. Available: https://www.boe.es/buscar/act.php?id=BOE-A-2018-16673

[24] "Morphcast emotion ai html5 sdk documentation," 2023, visitado el 01/06/23. [Online]. Available: https://ai-sdk.morphcast.com/latest/index.html

ANEXO

OBJETIVOS DE DESARROLLO SOSTENIBLE

Grado de relación del trabajo con los Objetivos de Desarrollo Sostenible (ODS).

Objetivos de Desarrollo Sostenible	**Alto**	**Medio**	**Bajo**	**No procede**
ODS 1. **Fin de la pobreza.**				X
ODS 2. **Hambre cero.**				X
ODS 3. **Salud y bienestar.**	X			
ODS 4. **Educación de calidad.**				X
ODS 5. **Igualdad de género.**			X	
ODS 6. **Agua limpia y saneamiento.**				X
ODS 7. **Energía asequible y no contaminante.**				X
ODS 8. **Trabajo decente y crecimiento económico.**				X
ODS 9. **Industria, innovación e infraestructuras.**				X
ODS 10. **Reducción de las desigualdades.**		X		
ODS 11. **Ciudades y comunidades sostenibles.**			X	
ODS 12. **Producción y consumo responsables.**				X
ODS 13. **Acción por el clima.**				X
ODS 14. **Vida submarina.**				X
ODS 15. **Vida de ecosistemas terrestres.**				X
ODS 16. **Paz, justicia e instituciones sólidas.**				X
ODS 17. **Alianzas para lograr objetivos.**				X

Reflexión sobre la relación del TFG/TFM con los ODS y con el/los ODS más relacionados.

La propuesta **Agenda 2030**, aprobada hace menos de una década, pone de manifiesto la voluntad de los gobiernos a tratar los problemas sociales surgidos en los últimos años a causa de diferentes factores, entre ellos la acción del ser humano sobre el entorno y sobre si mismo.
Los últimos años han supuesto para la sociedad un periodo donde percatarse de los problemas que asolan, junto a una profunda reflexión sobre los medios que disponemos para hacer frente. Estos problemas abarcan un amplio espectro, desde aquellos más asentados en nuestra historia (pobreza y hambruna), como aquellos más recientes y desconocidos para el ser humano (cambio climático y su influencia en la vida del planeta Tierra).
Nuestro trabajo también tiene su principal objetivo en la mejora de calidad de vida de los ciudadanos, dando solución a los problemas que afectan a estos. Parece ideal enmarcar nuestro trabajo dentro de la propuesta gubernamental para mostrar el potencial que tiene este. A continuación, detallaremos el razonamiento que hemos llevado a cabo para listar el grado de participación de nuestra solución en cada una de las propuestas de **ODS**:

1. **Salud y bienestar**: El objetivo principal y más claro de todo el trabajo. El proyecto se ideó para acercar la medicina y los tratamientos a los pacientes a modo de aumentar la cobertura sanitaria de los países. La salud es uno de los pilares del bienestar de los ciudadanos, y su calidad repercute de forma directa en la calidad de vida. Por ello, un sistema sanitario más conectado con sus usuarios eleva el número de diagnósticos, aumenta la calidad de tratamiento y mejora la percepción social sobre el estado de los servicios públicos de su propio país.

2. **Reducción de las desigualdades**: En la naturaleza de la solución se encuentra el ser una aplicación accesible para todo el mundo, compatible con dispositivos de prestaciones limitadas, pudiendo llegar tanto a las clases sociales más limitadas, como a países con acceso médico reducido.

3. **Igualdad de género**: Si bien la solución no responde ante una situación de desigualdad de género en específico, permite el acceso sanitario a los ciudadanos sin tener en cuenta el género del usuario. Además, las herramientas digitales ofrecen una capa de anonimato a quienes la emplean, eliminando los posible prejuicios y tabúes que puedan existir a la hora de buscar ayuda médica (en especial aquella de carácter psicológico por los estigmas sociales que envuelven la cuestión).

4. **Ciudades y comunidades sostenibles**: El hecho de haber creado la aplicación con funcionalidades limitadas (i.e. ausencia de animaciones cargantes, número de funciones disponibles limitadas, etc.) responde, entre otros, al objetivo de permitir su ejecución en dispositivos de prestaciones limitadas. De esta forma, se fomenta la reutilización de dispositivos antiguos en desuso para este propósito, así como evitar la necesidad de actualizar el dispositivo vigente del usuario.

ETS Enginyeria Informàtica
Camí de Vera, s/n, 46022, València
T +34 963 877 210
F +34 963 877 219
etsinf@upvnet.upv.es - www.inf.upv.es