POLITECNICO DI TORINO

Department of Computer Engineering DAUIN – Course LM/32
Computer Engineering - Cybersecurity

Master's thesis

“Machine Learning Foundations and Large Language Models for Digital Health Coaching: Market Study, Security Analysis and Prototype”

Thesis Supervisor: Maurizio Morisio
Candidate: Tirone Luca

Academic Year 2025-2026

Table Of content

Abstract

This thesis explores the landscape of digital health and wellbeing applications and reviews the theoretical foundations of machine learning, with a focus on convolutional neural networks and large language models. It compares state-of-the-art foundation models, GPT-4 and LLaMA-3, in terms of architecture, training strategies, and practical trade-offs. Cybersecurity challenges are examined in relation to health coaching apps that rely on Bluetooth-enabled wearables, with attention to standards, authentication mechanisms, and compliance with privacy frameworks such as GDPR and HIPAA.

Building on these analyses, the thesis presents the design of an AI-powered digital health coach. A central feature of the prototype is a lesson and quiz generation tool, which offers a platform for creating personalized educational content to support user engagement and experience. By combining market analysis, technical foundations, model comparisons, and security considerations, the work provides both a broad overview of the digital health domain and a concrete example of how advanced AI techniques can be applied in practice.

Chapter 1: Main competitors and their technologies

1.1 General Overview

The digital health monitoring market has evolved to include advanced tracking tools that not only monitor physical activity but also offer medically relevant insights. These two key elements combined with the accessibility provided by the "app" format have significantly broadened the adoption and the interest for the related technologies. Many applications today share core functionalities-such as monitoring heart rate, tracking sleep, and assessing overall wellness-but they diverge in terms of technology stacks, sensor integration, and machine learning (ML) implementations. These technical nuances result in diverse user experiences and market segmentation. This chapter explores some of the most prominent and thoroughly researched applications, with an additional focus on Bryan Johnson's Blueprint Program, detailing how their distinct technological choices affect their competitive standing in the market.

1.1.2 Market Overview

In 2023, the global market for health tracking applications was valued at approximately $50 billion and is projected to grow at a compound annual growth rate of 20% over the next five years. Key market drivers include the increasing prevalence of chronic diseases, advancements in wearable sensor technologies, innovations in AI-driven analytics, the integration of telemedicine services, and rising consumer concerns over data privacy. These drivers have fostered an environment of rapid innovation in platforms that deliver both fitness and medical insights.

1.2 Models and technologies

1.2.1 Machine Learning models and CNNs

Modern digital health and wellbeing applications are based on three core technologies (machine learning (ML), convolutional neural networks (CNNs), and large language models (LLMs)) to deliver a user-centric and fully automated experiences.
ML algorithms, including **supervised classifiers** that learn to detect health-related patterns from sensor and behavioral data, **unsupervised clustering** that groups users by lifestyle characteristics, and **reinforcement learners** that refine recommendations based on user feedback, function as the analytical backbone for predicting risks and producing and outputting advice based on individual goals and progress.
CNNs excel at interpreting visual inputs by automatically learning hierarchical features (from simple edges to complex shapes) through stacked convolutional and pooling layers allowing apps to estimate body-composition metrics or assess exercise form from user-captured photos and videos or even provide an accurate estimate of the calories of a meal without manual intervention.
Transformer-based LLMs leverage self-attention mechanisms to understand and generate coherent text in multi-turn conversations; when enhanced with retrieval-augmented generation (RAG), they dynamically pull in relevant facts from up-to-date knowledge bases, ensuring that guidance remains accurate and evidence-based.
Reinforcement learning from human feedback (RLHF) fine-tunes these language models by using expert annotations to reward safe, helpful, and contextually appropriate responses (an essential step for building user trust in health applications).
In conclusion the combination of ML analytics, CNN-driven vision, and LLM-powered dialogue form a unified, scalable framework capable of delivering data-driven insights, visual assessments, and natural-language coaching that adapt as users progress through their health, training, and wellbeing journeys.
The leading applications in these fields usually leverage a mix of proprietary machine-learning (ML) and large language models (LLM) to deliver a combination of accurate and personalized guidance.
Noom combines computer-vision based body-scan alongside a wellness assistant/companion named ("Welli") built on GPT-4 and Vertex AI with retrieval-augmented generation (RAG) for context-aware coaching.
Lark Health's AI Coach, prioritizing human-like interactions, makes use of supervised ML to gather data necessary to simulate empathetic, text-based counseling for weight and chronic-disease management.
Freeletics' Coach dynamically adjusts HIIT workouts via algorithms based on and fed with sports-science principles and user feedback, continuously refining plans based on the activity on their platform of millions of users.

MyFitnessPal's MealScan employs CNNs trained on millions of food images to automate the process of nutritional logging, while **Fitbod** uses supervised ML over 400million+ user data points to tailor strength-training regimens and optimize recovery.

For open-source LLM integration, LLaMA2, Falcon-40B, Mistral, and Vicuna emerge as top candidates, each offering distinct trade-offs in size, training corpus transparency, licensing, and performance.

1.2.2 Wearables

As displayed in the previous paragraph, it's noticeable how almost all the applications mentioned in this context heavily rely on data gathered through sensors often embedded in wearables such as waistbands, smartwatches and smart rings. Wearable devices integrate a suite of miniaturized sensors to continuously capture physiological and behavioral data, forming the foundation for personalized health and fitness insights.

Accelerometers and **gyroscopes** track movement and posture employed in common tasks such as step counting and activity classification.
Photoplethysmography (PPG) sensors use **light absorption variations** to monitor heart rate and blood-oxygen level, while **single-lead ECG modules** measure the heart's electrical activity to detect arrhythmias.
Skin-integrated temperature sensors record basal body temperature for illness detection and more frequently, metabolic assessments.
Barometers determine altitude changes to refine calorie-burn estimates during hikes, flights or dives.
GPS chips provide geo-location and speed data for outdoor workouts
Galvanic skin response (GSR) electrodes gauge sweat-induced skin conductance as a proxy for stress and emotional arousal.
SpO2 sensors estimate blood oxygen saturation via multi-wavelength PPG for respiratory and sleep monitoring.

Combined, these technologies provide a vast multimodal data stream feeding advanced analytics, ML-driven pattern detection, and real-time coaching algorithms.

1.2.3 Main competitors

The selection of the applications analyzed in this chapter is based on their strong alignment with the objectives of this thesis and their relative popularity, finally a particular attention has been paid to isolating competitors offering original technological solutions or approaches. Each application was chosen because it effectively demonstrates the integration of advanced technologies (such as ML algorithms, CNNs, and large language models) into user-centric platforms that enable tailored recommendations and human-like experiences. Moreover, these applications stand out for their proven ease of use and their **integration with wearables**, offering intuitive interfaces that foster user engagement and commitment. Their widespread popularity and significant presence in the market further validate their relevance not just for the larger sample of users but also as they represent an established benchmark within the rapidly evolving sector of AI-driven digital coaching and health monitoring.

1.3 Health monitoring/coaching applications

1.3.1 Noom

Noom focuses only on one aspect, nutrition, but the user-friendly approach together with an intuitive UI and the "AI-powered personal health assistant" makes a particularly interesting study case. Noom combines a classic coaching approach aimed at improving behavior and habits, with AI-powered features such as AI food logging, voice/text meal recognition (through computer vision), and "Welli," its AI personal health assistant, offering advice and habit formation support, specifically focus on establishing healthy habits more than just providing factual guidance.
Recent updates include an AI Body Scan module going beyond weight and BMI to monitor lean mass and fat percentage over time.

1.3.2 Lark Health

Lark's AI Health Coach monitors activity, weight, and vital signs, then mimics **empathetic counseling** in a text-based chatbot for weight management and chronic-disease prevention. While Lark mainly focuses on guidance for a healthy lifestyle for people suffering from chronic diseases, its comprehensive approach and the large **medical** dataset deployed underlying a scientific approach focused on disease control make it an interesting key player. Lark has also explored generative AI approaches (integrating clinician-guided medication strategies with behavior-change coaching) to augment its prevention and management programs for conditions like diabetes and obesity.

1.3.3 Freeletics

Freeletics uses AI-powered coaching algorithms to continuously personalize high-intensity interval training (HIIT) workouts, adjusting exercise selection, sets, and repetitions based on real-time user feedback and performance metrics. The Freeletics Coach employs data-driven learning to estimate abilities and refine recommendations, tailoring training plans to individual profiles by considering factors such as age, gender, and fitness level. With the recent launch of Coach+, Freeletics has enhanced its AI modules for deeper personalization and motivational support, integrating both algorithmic adjustments and optional live coach interactions for clarification and guidance providing an interesting hybrid approach to the deployment of AI coaching suits.

1.3.4 MyFitnessPal

MyFitnessPal is a nutrition and activity-tracking platform that helps users monitor calorie intake, macronutrients, and exercise. Logging is handled both manually and with semi-automated options such as food search, barcode scanning, and community-shared recipes. Behind the scenes, traditional ML models learn from each person’s history to surface likely foods, speed up entry with predictive suggestions, and adjust daily calorie and macronutrient targets in line with weight trends and stated goals.
The app also offers **MealScan**, which analyzes smartphone photos to recognize dishes and prefill nutrition fields. More recently, MyFitnessPal has begun rolling out AI-assisted meal planning that assembles weekly menus from a user’s preferences and constraints.

1.3.5 Fitbod

Fitbod is designed to deliver personalized strength-training workout programs designed to adapt to each user's goals, available equipment, and performance history. Through an intuitive UI the user can select intensity, training frequency and targeted muscle groups and receive dynamically generated sessions that balance exercise variety with progressive overload principles. By continuously tracking completed workouts, logged weights, and user feedback, Fitbod ensures gradual improvement, making comprehensive strength training accessible to both novices and experienced lifters.
Fitbod uses machine-learning algorithms trained on 400million+ data points to generate customized strength-training workouts that adapt to each user's equipment, performance history, and recovery status. Its "AI trainer" places particular emphasis on continuously refining exercise selection and volume to optimize progress while **minimizing injury risk**, making it a leading ML-driven fitness planner in its own field.

1.3.6 WHOOP

WHOOP is a wearable fitness monitor app, designed to pair with a specific wristband and continuously collect biometric data (such as heart rate variability, resting heart rate, respiratory rate, and sleep metrics). The collected data is then translated into daily Recovery, Strain, and Sleep scores. Whoop's strong suit resides in their proprietary algorithms calibrated against both individual and population baselines. Its Health Monitor feature streams live heart rate, SpO_2, and skin temperature metrics to the companion app, alerting users when values deviate from their personalized norms. Clinical validation studies report WHOOP's acceptable accuracy for two-stage sleep classification and heart rate tracking. By cross-referencing data (HRV, RHR, sleep quality, and respiratory rate) coming from their sensors WHOOP delivers continuous insights on readiness and recovery (WHOOP supports data-driven training load adjustments and recovery strategies, making it especially popular among elite athletes and performance enthusiasts.

1.3.7 Google Fit (data sharing and aggregation)

Google Fit adopts a cloud-centric strategy based on TensorFlow and Google Cloud AI to process large volumes of biometric data (including physical activity, heart rate, and respiratory metrics) collected from a large variety of sensors. Its approach to sensor fusion, which combines data from accelerometers, gyroscopes, and optical sensors, provides a relatively high level of accuracy of activity recognition and health predictions. The open and flexible ecosystem of Google Fit favors integration with a large selection of wearables and third-party applications, facilitating scalability and continuous innovation. The open ecosystem simplifies third-party integration and cross-platform integration, though reliance on cloud processing can raise privacy considerations and produce device-to-device variability.

1.3.8 Apple Health (data sharing and aggregation)

Apple Health is a health data aggregation platform developed by Apple Inc., designed to centralize and track a broad range of wellness metrics from both Apple devices (such as the iPhone and Apple Watch) and third-party apps and wearables. The platform organizes user data into key categories such as activity, sleep, heart health, mindfulness, nutrition, and vitals, providing detailed visualizations and trend summaries over time in a particularly intuitive and user-friendly format. Apple Health emphasizes user privacy and transparency, granting individuals full control over which apps and devices can access their data. Apple's

application strong suite resides once more in its comprehensive approach and its user-friendliness, the dashboard-like approach provides an easy-to-process format to inform the user of their health and wellness.

1.4 Overview of deployed Technologies per competitor

1.4.1 Noom

Noom provides an on-app Body Scan using a convolutional neural-network pipeline to analyze a 10-second user video and compute body-composition metrics (body fat, lean mass, waist-to-hip ratio) from proprietary data, additionally it offer an AI based assistant, Welli. Initially prototyped on GPT-4, then extended to leverage both OpenAI and Google Vertex AI models with RAG built and trained based on Noom's internal knowledge base, Welli offers personalized guidance and human-like interactions.

1.4.2 Lark Health

Lark's AI Health Coach ingests smartphone sensor streams (activity, weight, vital signs) and integrated device data, applying supervised ML models to detect health patterns and generate automated, empathetic chat-based interventions for weight management and diabetes prevention.

1.4.3 Freeletics

The Freeletics Coach employs a rules-and-data-driven algorithm that adjusts HIIT workout plans in real time based on user feedback, completion rates, and sports-science parameters, having been trained on behavioral data from over 56million users to calibrate difficulty, volume, and recovery recommendations.

1.4.4 MyFitnessPal

MyFitnessPal combines a crowdsourced database of over 14 million food entries with supervised machine learning to detect individual eating patterns and offer predictive autocomplete suggestions for frequently consumed items, significantly streamlining manual logging. The MealScan feature employs convolutional neural networks trained on millions of labeled food images combined with computer vision elements to automatically recognize dishes from smartphone photos and populate detailed nutritional fields with minimal user effort. Additionally, MyFitnessPal provides information on daily calorie and macronutrient using established equations such as Mifflin-St Jeor to estimate basal metabolic rate, dynamically adjusted via simple predictive models that factor in real-time activity data and ongoing weight trends. Lastly MyFitnessPal can also leverage generative AI meal-planning engines (acquired from specialized providers) to algorithmically construct balanced, goal-aligned weekly menus and corresponding grocery lists tailored to users' dietary preferences, allergies, and budgets.

1.4.5 Fitbod

Fitbod's workout engine applies supervised learning on over 400million user workout logs, leveraging collaborative-filtering and regression techniques to predict optimal exercises, sets, and reps based on equipment availability, user ability, and recovery metrics.

App	Primary Goal	Key AI / ML Technologies	Main Data Sources
Noom	Behavior-change nutrition coaching	CNN body-scan analysis; GPT-4-based LLM assistant *Welli* with RAG	User body-scan videos; food logs
Lark Health	Chronic-disease lifestyle coaching	Supervised ML on sensor streams; generative-AI text chatbot	Smartphone & wearable sensor data
Freeletics	AI-personalized HIIT training	Hybrid rule-based + supervised ML workout engine	56 million user workout histories
MyFitnessPal	Comprehensive nutrition & fitness logging	CNN *MealScan* vision; predictive autocomplete; generative meal planner	14 million-item food DB; user food photos
Fitbod	AI-personalized strength training	Collaborative-filtering & regression models on 400 M logs	User workout logs; connected-device data
WHOOP	Recovery & strain monitoring	LSTM time-series models on HRV/sleep; edge analytics on wearable	PPG, accelerometer, skin-temp sensors
Google Fit	Aggregated health data & activity tracking	TensorFlow cloud models; multi-sensor fusion algorithms	Accelerometer, gyroscope, optical HR data
Apple Health	Aggregated health data & activity tracking	Machine learning algorithms applied to large datasets collected from sensors	ML, CNNs, and accelerometer, skin temps, sensors, HRV

Chapter 2: Theoretical Foundations

2.1 Introduction

In the past decade, artificial intelligence (AI) has progressed from a research concept to a practical tool driving innovation across industries. Some of the main technologies core to this transformation are: machine learning (ML), convolutional neural networks (CNNs), and large language models (LLMs). This chapter provides a comprehensive but accessible overview of each of these technologies. We will discuss how they work, what they require, their capabilities and limitations, and why they have become essential tools in modern computing, especially in domains like health and personalized coaching.

2.2 Machine Learning: A Foundational Concept

Machine learning (ML) is a field that focuses on developing algorithms capable of learning from data and making predictions or decisions with minimal human intervention. It's generally possible to distinguish two main categories of machine learning: supervised and unsupervised learning.

ML enables systems to learn patterns from data without being explicitly programmed for each specific task. In supervised learning, models are trained on labeled datasets (meaning each training example includes both an input and a correct output). The goal is for the model to learn a general mapping from inputs to outputs, allowing it to predict outcomes for new, unseen data. For example, a supervised model might learn to classify email messages as spam or not spam based on past examples.

In contrast, unsupervised learning deals with unlabeled data. These algorithms look for hidden structures or patterns **without** relying on predefined outcomes.

Techniques like **clustering** and **dimensionality reduction** fall into this category. Clustering algorithms group similar data points together, while dimensionality reduction methods like Principal Component Analysis (PCA) aim to simplify high-dimensional data into more manageable forms.

Finally, hybrid approaches exist too. **Semi-supervised** learning uses a small amount of labeled data combined with a larger pool of unlabeled data, while **reinforcement learning** allows models to learn optimal actions through trial-and-error, often used in robotics and game environments.

Training machine learning models generally consists in minimizing a loss function, which quantifies the difference between the model's predictions and the factual outcomes. Algorithms are employed, like gradient descent, to iteratively adjust model parameters to reduce this error. To avoid overfitting (where the model performs well on training data but poorly on new data), techniques like regularization, dropout, and early stopping are applied.

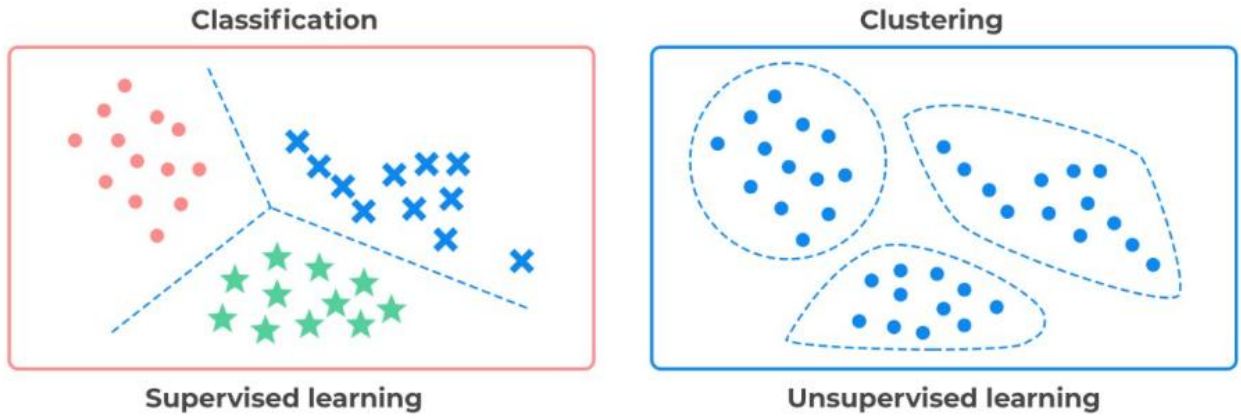

Fig. 1 Conceptual difference between supervised and unsupervised learning

ML is a wide term and it covers a wide range of algorithm families, including linear models, decision trees, support vector machines, ensemble methods (like random forests), and neural networks. Each comes with its own assumptions and strengths, making them suitable for different tasks and data types.

Modern ML applications are data-hungry as the performance of these models often scales with the amount and quality of data available. At the same time, the computational demands of training have risen, making GPUs and other accelerators a necessity for large-scale ML tasks.

2.3 Convolutional Neural Networks (CNNs)

Convolutional Neural Networks, or CNNs, are a specialized type of neural network particularly effective for processing grid-like data, such as images. They are widely regarded as the cornerstone of modern computer vision.

At a high level, CNNs work by learning spatial hierarchies of features. This means that the network learns low-level features like edges in the first layers and more complex structures like shapes or objects in the deeper layers. This is made possible by convolutional layers, which apply learnable filters across the input data to detect patterns.

The output of these convolutions is passed through a nonlinear activation function, commonly ReLU (Rectified Linear Unit), which introduces nonlinearity into the model. This is followed by pooling layers that downsample the data, reducing the spatial dimensions and computation requirements while preserving important features.

The final stages of a CNN typically include fully connected layers, where the extracted features are used to make predictions. These might be class labels in the case of image classification, or coordinates in the case of object detection.

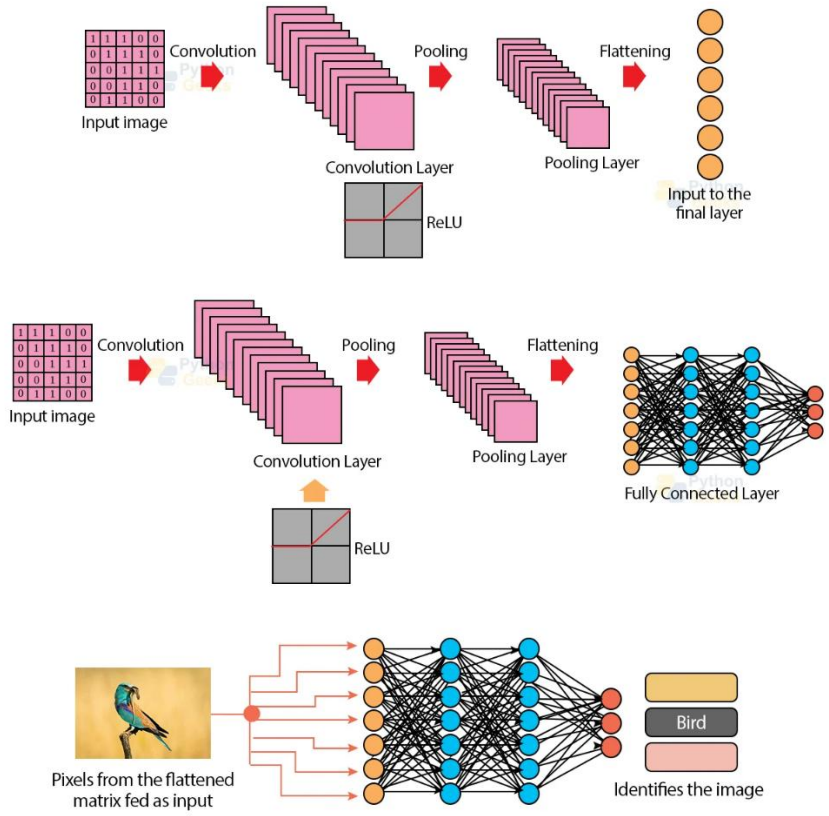

Fig. 2 main steps of the activity of a CNN.

A key advantage of convolutional neural networks lies in how they handle parameters. Instead of assigning separate weights to every connection, as in fully connected layers, CNNs reuse the same filters across the spatial dimension. Using the same weights greatly reduces the total number of parameters and helps the model generalize better.

As mentioned before, training CNNs still depends on access to large, labeled datasets and significant computational resources. The ImageNet benchmark, containing more than 14 million labeled images, has been especially important for progress in the field, enabling the development of well-known CNN architectures such as AlexNet, VGG, ResNet, and

Inception.
While CNNs are generally used for image-related tasks, they have also been adapted for time-series data, audio processing, and even some text-based applications, although the latter are now more commonly handled by Transformer-based models such as GPT.

2.4 Large Language Models (LLMs)

Large Language Models represent a significant step forward in AI, enabling machines to generate and understand human-like text with remarkable fluency. These models are built upon the Transformer architecture, introduced in 2017, which moved away from earlier models based on recurrence (like RNNs) and convolutions.
The Transformer relies on multi-head self-attention, which allows it to analyze relationships between all words in a sentence simultaneously, rather than sequentially. This enables greater parallelization during training and allows the model to capture complex dependencies, even across long passages. Each layer in a Transformer includes both a self-attention mechanism and a feed-forward neural network, with residual connections and layer normalization to stabilize and accelerate learning.

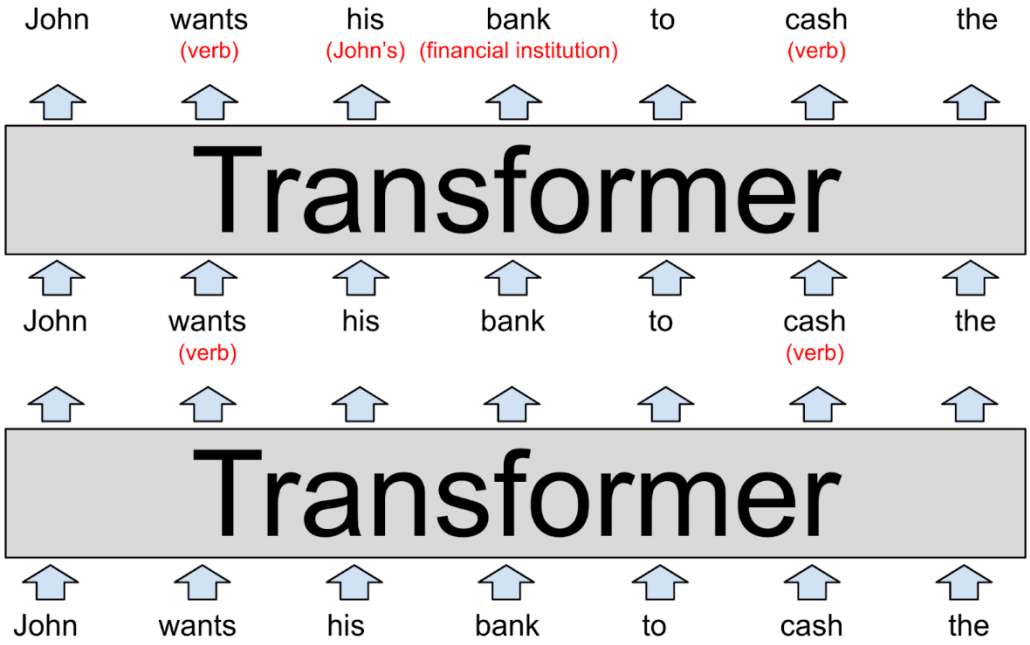

Fig. 3 Conceptual functioning of a transformer.

In contrast to earlier neural network models, Transformers are not limited to using a fixed context window. To keep track of the order of the tokens in a sequence, they attach to the input embeddings positional information, known as **positional encodings**. These architectural choices make Transformers highly scalable, efficient to train on modern hardware, and capable of learning intricate language patterns at scale.
The core idea behind LLMs is the concept of **self-attention mechanisms**, this technology allows the model to weigh the importance of different words in a sentence relative to each other, regardless of their position. This allows the capture of long-range dependencies and subsequently the contextual relationships, something previous models struggled with.
At a more fundamental level, LLMs operate on word vectors, where each word or token is represented as a high-dimensional vector in a continuous space. These embeddings capture semantic properties-such as similarity or analogy, enabling the model to reason about

language numerically. During training, these word vectors are refined to reflect contextual meaning and relationships across vast a wide spectrum of content.

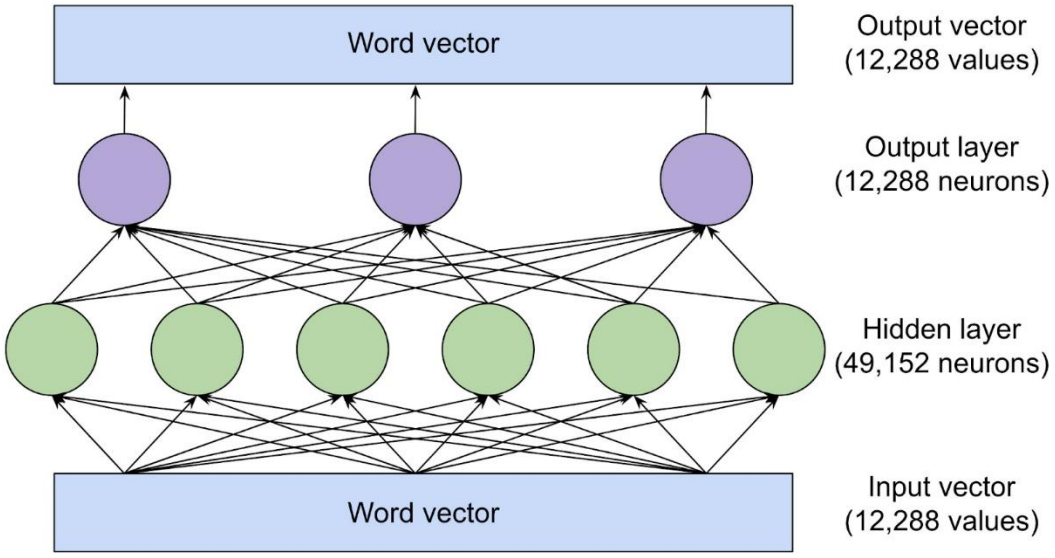

Fig. 4 conceptual representation of word vectors.

LLMs are trained on massive amounts of text data (from books and articles to internet itself) to learn to predict the next word in a sentence given the previous words and their relationships. This simple training task enables complex capabilities such as summarizing documents, translating languages or even writing code and answering questions.
Training LLMs requires extraordinary amounts of data and computational resources. Models like GPT-3 and GPT-4 and more recently, GPT-5, contain hundreds of billions of parameters and are trained on hundreds of billions of words. Training them requires the usage of specific computing efforts, carried out by thousands of GPUs over weeks or at times even months.
To improve the safety, reliability and usefulness of LLMs, techniques like Reinforcement Learning from Human Feedback (RLHF) and Retrieval-Augmented Generation (RAG) are employed.
RLHF refines a pre-trained model by exposing it to human-generated examples of preferred outputs and using reinforcement learning algorithms (such as Proximal Policy Optimization (PPO)) to reward responses that align with human judgment, improving helpfulness, coherence, and safety.
RAG, in contrast, enhances factual accuracy by integrating a retrieval system: the model dynamically searches an external database or document collection during inference and incorporates the retrieved content into its response. This hybrid approach combines the generative power of LLMs with the factual precision of a structured knowledge base.
LLMs have found wide applications in both general and domain-specific contexts. In health technology, for instance, they can be used to generate personalized coaching messages, summarize patient data, or even interpret medical research literature.

2.5 Summary Table: Technology Comparison

Technology	Input Type	Key Strengths	Limitations	Common Applications

ML	Structured data	Versatile, interpretable (in simple models), efficient for tabular data	Needs labeled data, risk of overfitting	Classification, regression
CNN	Image, grid data	Excellent at spatial pattern recognition, efficient parameter sharing	Data-hungry, limited long-range dependency modeling	Image recognition, medical imaging
LLM	Text sequences	Contextual understanding, generative capabilities, few-shot learning	High computational cost, risk of hallucination	Chatbots, translation, summarization

2.6 Conclusion

Machine learning, convolutional neural networks, and large language models each represent distinct approaches to creating a form of intelligent systems each relevant in slightly different context and applications. ML provides the foundation for data-driven learning, CNNs specialize in visual and spatial understanding, and LLMs push the boundaries of what machines can do with natural language and human-like interactions. Continuously improving and revolutionizing the development and application these tools, not only helps explain their current success, but also prepares us to better leverage them in emerging applications, such as digital health management, where personalized, adaptive support is increasingly vital.

Chapter 3: GPT-4 vs LLaMA-3 Analysis

3.1 Introduction

The progression of large language models (LLMs) in recent years has been marked by rapid innovation, largely driven by Transformer-based architectures applied to vast and diverse text corpora. OpenAI's Generative Pre-trained Transformer (GPT) series, starting with GPT-1 up to GPT-4 and more recently GPT-5 has played a pivotal role in this transformation, with each iteration significantly expanding both model size, capabilities and thus popularity.

Presently many other competitors have stepped in, most notably Meta introduced the LLaMA (Large Language Model Meta AI) series, focused on delivering efficient and **openly available** foundation models. LLaMA-1, launched in 2023, proved that compact models ranging from 7 billion to 65 billion parameters could match or even exceed the performance of much larger systems when trained on carefully curated **public** datasets. LLaMA-2, released the same year, introduced models up to 70 billion parameters, including fine-tuned chat variants. By 2024, Meta unveiled LLaMA-3, a suite of models highlighted by a dense 405-billion-parameter version and the introduction of ultra-long context windows.
GPT-4, released in 2023, similarly represented a leap forward, surpassing GPT-3 in both scale and capability and functionalities. Notably, it incorporated multimodal functionality

(accepting both text and images as input) and benefited from extensive human feedback during fine-tuning.
Both GPT and LLaMA model families illustrate the prevailing paradigm: large-scale pretraining of transformer models followed by alignment to human tasks through supervised or reinforcement-based fine-tuning. These systems now serve as multipurpose tools capable of tasks such as natural language generation, translation, summarization, and programming assistance.
It's worth noting that while GPT models have primarily been developed for commercial use and therefore emphasizing the aspects of performance and safety the LLaMA models have prioritized openness and research usability. Meta has released the code and weights for its models under a community license, facilitating public access and modification.
This chapter offers a comprehensive analysis of the GPT-4 and LLaMA-3 models by comparing key elements such as architecture, training methodology, data sources, tokenization approaches, performance metrics, and practical trade-offs. Notably, due to LLaMA's open-source nature, available technical details are more extensive and accurate making them particularly valuable in the context of academic scrutiny.
On the other hand, numerical data referring to GPT models presented in this chapter is to be considered speculative and resulting from industry reports and blog sources.

3.2 Fine-Tuning a Large Language Model

The concept of fine-tuning refers to the targeted adaptation of a pre-trained large language model (LLM) to a specific domain, task, or application. This process provides an additional training on a smaller, more specialized dataset, which enables the model to align more closely with particular goals. While the original pre-training phase exposes models such as the afore mentioned GPT-4 or LLaMA-3 to broad, heterogeneous text corpora, fine-tuning narrows this focus, allowing the model to internalize specific formats, jargon, and reasoning patterns.
Typically, fine-tuning is conducted via **supervised learning**. Here, the model is presented with pairs of inputs and desired outputs, and its internal weights are iteratively adjusted using gradient descent to reduce prediction error. This methodology enables the model to grasp nuanced relationships within a specific context, this process effectively increases the precision and relevance of the output.
In addition to standard supervised fine-tuning, developers may utilize more advanced strategies. Instruction tuning, for example, involves training a model to follow structured prompts effectively. Another prominent technique is Reinforcement Learning from Human Feedback (RLHF), which adjusts model behavior to align more closely with human preferences by leveraging evaluative feedback during training. A third approach is continued pre-training, where a model is further exposed to in-domain text in an unsupervised fashion, improving its familiarity with subject-specific content.
Crucially, these methods enable organizations to repurpose general-purpose LLMs for custom applications without incurring the immense computational costs associated with full-scale model training from scratch.

3.2.1 GPT-4 Architecture and Training

GPT-4 represents a significant step beyond GPT-3 in both architecture and computing. Although OpenAI has not publicly disclosed many specifics, external analyses and industry reports suggest that GPT-4 relies approximately 1.8 trillion parameters distributed across roughly 120 transformer layers. According to the same industry reports, GPT-4 also

introduces uses a sparse Mixture-of-Experts (MoE) framework, which differently from traditional dense architectures activates only a subset of the model's sub-networks during each forward pass.
Currently available information suggests GPT-4 utilizes around 16 distinct expert modules, each relying on about 111 billion parameters. During inference, only two experts are active for any given input. This setup allows GPT-4 to achieve a larger total parameter count while maintaining a relatively small time footprint similar to that of much smaller dense models. In practical terms, GPT-4 requires approximately three times the compute resources of GPT-3's largest 175B model during inference, despite its vastly increased capacity.
Additionally GPT-4 is a multimodal model, capable of processing both textual and visual inputs. OpenAI has stated that the model integrates a vision encoder module, allowing it to interpret image data before passing it to the core language model. This capability was achieved through further fine-tuning using an additional two trillion training tokens, many of which consisted of paired image-caption datasets.

The training of GPT-4 consisted of a two-stage process.
The initial phase involved autoregressive next-token prediction using a vast unlabeled dataset. This corpus is believed to contain around 13 trillion tokens and includes content drawn from web crawls (such as Common Crawl and RefinedWeb), programming code, and curated textual resources. This represents an order of magnitude more training data than that used for GPT-3, which was trained on roughly 300 billion tokens.
For the second part OpenAI invested significant effort into alignment and safety fine-tuning. For approximately six months, researchers iteratively refined the model's outputs using human evaluations, preference data, and adversarial testing a process known as RLHF. These efforts led to substantial improvements: according to OpenAI, GPT-4 is 82% less likely to produce disallowed content and 40% more likely to return factual responses than GPT-3.5. Nonetheless, GPT-4 continues to exhibit common LLM limitations, including susceptibility to hallucinations, retention of training data biases, and the inability to update its knowledge dynamically post-training.

3.2.2 LLaMA-3 Architecture and Training

Meta's **LLaMA-3 family**, sometimes referred to as the "Herd of Models," includes a range of open-source foundation models. The largest is a dense Transformer of around **405 billion parameters**, a significant step up from the **70 billion parameters** of LLaMA-2. A defining feature of LLaMA-3 is its extended **context window**, which in certain variants reaches **128,000 tokens**, making it one of the first open-source models to operate at that scale.

Unlike GPT-4, which is reported to employ a sparsely activated Mixture-of-Experts design, LLaMA-3 relies on a **fully dense architecture**: all parameters are active at inference time. This design favors consistency across inputs, though Meta emphasizes that much of the improvement from LLaMA-3 derives from better data quality and larger scale rather than entirely new architectures.
Several refinements distinguish the series:

1. **Grouped-Query Attention (GQA):** reduces memory use during decoding by sharing key/value heads (eight per attention block in LLaMA-3).**Document-Scoped Attention Masks:**
2. **Document-scoped attention masks:** constrain attention to tokens within the same document when processing long sequences, which preserves contextual integrity.

3. **Expanded tokenizer vocabulary:** the vocabulary now contains 128k tokens. Reports suggest that about 100k tokens are inherited from OpenAI's *tiktoken* system, with the rest added to improve coverage of non-English languages. This design yields greater compression in English (roughly 3.9 characters per token compared to 3.2 in LLaMA-2).
4. **Extended Positional Embeddings:**
   by adjusting the base frequency of Rotary Positional Embeddings (RoPE), the models natively handle sequences of ~32k tokens during pre-training and can be fine-tuned up to 128k.
5. **Model scale:** the 405B variant uses 126 layers with a hidden size of 16,384 and 128 attention heads, compared to 80 layers and 8,192 hidden size in LLaMA-2's largest release.

LLaMA-3 models are released in both base (pretrained) and "instruct" variants. These instruct models are fine-tuned using supervised instruction-following datasets as well as reinforcement learning techniques. As of July 2024, Meta has released 8B and 70B instruct models, in addition to larger 405B variants. The LLaMA-3 series also introduces Llama Guard 3, a safety-focused model designed to filter potentially harmful inputs and outputs. This extensive ecosystem positions LLaMA-3 as a flexible and adaptable platform for research and deployment.

3.2.3 Training Data

The pre-training of LLaMA-3 involved approximately 15 trillion tokens, a substantial increase compared to the ~2 trillion tokens used for LLaMA-2. Meta placed a strong emphasis on data quality, applying rigorous filtering to eliminate low-value, redundant, or inappropriate content, such as personal information and explicit material.
The dataset composition for LLaMA-3 is as follows:

1. 50% general knowledge - sourced from high-quality web content, books, and encyclopedic text;
2. 25% mathematics and reasoning-related material - to boost problem-solving and logical inference capabilities;
3. 17% programming code - to improve coding tasks;
4. 8% multilingual data - covering a broad spectrum of non-English languages.

Training was carried out in multiple phases. The first stage utilized a context length of 8,000 tokens. Subsequently, a continued pre-training phase extended the model's capacity to handle sequences up to 128,000 tokens. This staged strategy allowed the model to learn efficiently while gradually increasing its ability to manage longer contexts.

3.2.4 Compute and Parallelism

Training the LLaMA-3 405B model required not only massive computational resources but also a highly specialized infrastructure. Meta relied on its **"Grand Teton" AI supercluster**, a purpose-built system that can host up to **24,000** NVIDIA H100 **GPUs**. For the training of LLaMA-3, as many as **16,000** H100 units were used, each equipped with 80 GB of HBM3 **memory** and rated at a 700-watt TDP.
To make such large-scale training feasible, Meta applied a strategy known as **4D parallelism**, which combines four complementary forms of parallel computation:

1. Tensor parallelism: the operations within each layer are split across multiple GPUs, so

that large matrix multiplications can be executed in parallel rather than on a single device.
2. Pipeline parallelism: assigning different parts of the model to distinct GPUs;
3. Data parallelism: training batches are distributed across groups of GPUs, each processing a portion of the data and then synchronizing gradients. This is the most common form of parallelism in deep learning.
4. Expert parallelism: used for components that incorporate **Mixture-of-Experts** structures, such as auxiliary systems like *Llama Guard*. Different experts are assigned to different GPUs, enabling the model to scale efficiently without every GPU computing every expert's parameters.

Inference with LLaMA-3 also benefits from optimized hardware configurations. For example, Meta uses FP8 (8-bit floating point) quantization and micro-batching techniques to improve throughput. These optimizations make it possible to serve even the massive 405B model efficiently-albeit still requiring powerful distributed systems.

3.2.5 Tokenization for GPT

Tokenization plays a critical role in the performance of language models, as it determines how input text is broken down into tokens that the model can process. GPT models historically utilize a byte-level Byte Pair Encoding (BPE) tokenizer. GPT-3's tokenizer used around 50,000 tokens, while GPT-4-though undocumented in precise detail-employs a system consistent with OpenAI's tiktoken library.
LLaMA, by comparison, uses a UTF-8 byte-level BPE tokenizer. In LLaMA-3, the vocabulary was expanded significantly to 128,000 tokens. This included the 100K base from tiktoken and an additional 28K tokens tailored to high-frequency terms in non-English languages.
This expanded tokenizer provides several benefits:

1. Improved compression: More text can be encoded using fewer tokens.
2. Enhanced multilingual capability: Non-English texts are handled more efficiently.
3. Minimal regression for English: Despite the broader token space, performance on English tasks remains strong.

Differences in tokenization design between GPT and LLaMA affect how each model handles compression, vocabulary diversity, and multilingual generalization.

3.2.6 Key Advances Over Earlier Versions: From GPT-2/3 to GPT-4

The developmental arc of OpenAI's GPT series is characterized by dramatic increases in scale and performance. GPT-2, released in 2019, featured 1.5 billion parameters and demonstrated surprising capabilities in text generation. GPT-3, launched in 2020, took a major leap to 175 billion parameters. Despite retaining a dense architecture, GPT-3 introduced the concept of in-context learning, showing that large models could perform a variety of tasks using only a few example prompts-without the need for additional gradient updates.
GPT-3.5, released in 2022, offered a refined version of GPT-3, largely based on the results achieved through alignment techniques such as reinforcement learning from human feedback (RLHF). Although specific details remain sparse, it served as a bridge between GPT-3 and the significantly more powerful GPT-4.
With the release of GPT-4 in 2023, OpenAI introduced several key advances:

1. Scale: At an estimated 1.8 trillion parameters, GPT-4 is approximately 10 times larger than GPT-3.
2. Sparse Activation: Its Mixture-of-Experts architecture enables scalable training and

inference without linear increases in compute.
3. Multimodal Inputs: GPT-4 integrates vision capabilities, allowing it to interpret and respond to image inputs.
4. Extensive Training Data: The training dataset consisted of about 13 trillion tokens, far exceeding the datasets used for earlier models.

These improvements provided a significant real-world performance gain, for instance, GPT-4 ranked in the top 10% of test takers on the U.S. bar exam, whereas GPT-3.5 had scored in the bottom 10%.
Benchmarks showed consistent improvements across mathematical reasoning, common sense, language comprehension, and code generation.
Additionally, GPT-4 demonstrated increased robustness and broader multilingual support. On the MMLU benchmark (a comprehensive benchmark designed to evaluate the knowledge and problem-solving abilities of large language models (LLMs)), GPT-4 often surpassed English-language state-of-the-art models in dozens of other languages, reinforcing the impact of scale and architectural refinement.

3.3 From LLaMA 1/2 to 3

Meta's LLaMA series has advanced rapidly since its initial release in early 2023 to the current, more sophisticated models in the LLaMA-3 family. LLaMA-1 was introduced in February 2023, offered models ranging from 7 to 65 billion parameters, their exceptional performance demonstrated that high-quality, smaller-scale architectures could rival or even surpass much larger systems, provided a high-quality training on curated public datasets.
Later that year, in July 2023, the second iteration of LLaMa, LLaMA-2 was released. It offered a larger set of parameters up to 70 billion and introduced significant enhancements, including publicly available base and instruction-tuned chat variants. One of the key improvements in LLaMA-2 was the extension of the context window to 4,096 tokens doubling that of its predecessor. Additionally, grouped-query attention (GQA) was integrated into the largest models to optimize memory and speed during inference.
LLaMA-3 continued on the same trajectory, introducing a 405-billion-parameter model with a context window reaching up to 128,000 tokens. LLaMA-3 also underwent an optimization process aimed at improving its throughput, multilingualism, and tokenizer efficiency. Meta's internal evaluations indicate that LLaMA-3's performance is on par with GPT-4 across many domains, including math, reasoning, and language understanding. Furthermore, the release of instruction-tuned variants has enhanced the model's usability for prompt-based applications, positioning LLaMA-3 as a competitive **open** alternative to proprietary systems.

3.3.1 Training Datasets and Tokenization

GPT-4 and LLaMA-3 differ notably in the scale and transparency of their training datasets. GPT-4 is reported to have been trained on approximately 13 trillion tokens drawn from a broad selection of sources including web pages, books, encyclopedic entries, and publicly available code. However, OpenAI has not fully disclosed the composition or curation methods of this dataset.
LLaMA-3, by contrast, was trained on approximately 15.6 trillion tokens. Meta has provided more details about the dataset structure, which includes:

1. A significant proportion of web and book data (general knowledge).
2. Substantial representation of programming and reasoning content.
3. A multilingual subset representing roughly 8% of the total dataset.

Meta's strategy involved strict quality controls, filtering out content considered low-value or inappropriate, such as personal data or explicit materials and on the other hand, topics like mathematics and coding were deliberately upweighted to improve performance in specialized tasks.
Both GPT-4 and LLaMA-3 use **subword tokenization methods**, where words are split into smaller units to help the model handle rare or unknown words more effectively while keeping the vocabulary size manageable. GPT-4 utilizes OpenAI's byte-level BPE tokenizer (as seen in GPT-3's tiktoken implementation), whereas LLaMA-3 uses an enhanced version of BPE with a 128,000-token vocabulary. This expansion allows LLaMA-3 to encode non-English content more efficiently and compress text more effectively overall.
The impact of these tokenization choices is significant. LLaMA-3's broader vocabulary allows it to represent more information per token, especially in non-English languages, improving model efficiency and multilingual accuracy.

3.4 Infrastructure Requirements

The computational demands associated with training models like GPT-4 and LLaMA-3 are immense. GPT-4 was trained on an AI custom built supercomputing infrastructures provided by Microsoft’s Azure. According to OpenAI, the total cost of training GPT-4 reached approximately $63 million. The model was deployed using clusters comprising around 128 of the most powerful GPUs available at the time, operating with complex parallelization strategies that include tensor and pipeline parallelism.
LLaMA-3, in contrast, was trained using Meta's proprietary "Grand Teton" AI supercluster, an internal infrastructure specifically design designed to support large-scale model training with high bandwidth and extensive GPU interconnectivity. Up to 16,000 NVIDIA H100 GPUs, each offering 80GB of HBM3 memory, were employed in training the 405B LLaMA-3 model. These GPUs were distributed across servers linked via NVLink and a high-throughput RoCE networking fabric. The full cluster can support up to 24,000 GPUs.
As mentioned before, both models utilized 4D parallelism, a strategy that combines:

1. Tensor parallelism (dividing layer computations across devices),
2. Pipeline parallelism (distributing layers across GPUs),
3. Data parallelism (spreading data batches across nodes), and
4. Expert parallelism, used selectively in safety models like LLaMA Guard.

In deployment, GPT-4 runs exclusively through OpenAI-managed infrastructure, making it inaccessible for direct hosting. LLaMA-3, however, is open-source and can be deployed on private hardware, although full-scale use of the 405B variant still requires powerful multi-GPU systems.

Fig. 5 Meta’s Grand Teton

3.4.1 Licensing Models

One of the most significant differences between GPT-4 and LLaMA-3 lies in their licensing frameworks.

GPT-4 is a fully proprietary model. OpenAI has not released the model's architecture, training code, or weights. Access is limited to API usage through OpenAI's commercial services, including ChatGPT Plus and enterprise integrations. Users cannot host, inspect, or modify the model independently. This restriction prioritizes safety, consistency, and business control, but limits transparency and academic exploration.

In contrast, LLaMA-3 is distributed under Meta's LLaMA Community License. This license allows users to:

1. Download, reproduce, and distribute the models;
2. Create derivative works and fine-tuned variants;
3. Use the models for commercial and academic applications (subject to certain conditions).

However, the license does impose limitations. For instance, companies with more than 700 million monthly users must obtain separate commercial licensing. Additionally, the license prohibits using LLaMA models to develop or train competing large language models. Redistribution must include attribution statements such as "Built with Meta LLaMA 3."

These terms position LLaMA-3 as a broadly accessible research tool, with a framework that balances openness with safeguards for Meta's commercial interests.

3.5 Performance: Benchmarks and Behavior

3.5.1 Accuracy

When it comes to performance, GPT-4 has set new standards for accuracy across a wide range of benchmarks. According to OpenAI's internal testing, GPT-4 achieved an 86.4% score on the English MMLU (Massive Multitask Language Understanding) benchmark using five-shot prompting. This is a notable improvement over GPT-3.5, which scored only 70.7% on the same test.

GPT-4 also seems to excel in more focused domains. On the GSM-8K benchmark, which assesses the level of mathematical reasoning, GPT-4 reached a score of 92.0%, compared to GPT-3.5's 57.1%. It also outperformed predecessors on commonsense and reasoning tasks, achieving 95.3% on HellaSwag.

LLaMA-3, particularly its 405B model, performs competitively with GPT-4 across many of these benchmarks. Meta's evaluation indicates that LLaMA-3 nearly matches GPT-4 in few-shot settings across math, reasoning, and language understanding. On the HumanEval benchmark, which tests code generation, GPT-4 achieved a rate of 67%, compared to LLaMA-3's 47%.

While GPT-4 maintains an edge in programming-related tasks, LLaMA-3 is generally on par in general NLP evaluations. Smaller LLaMA-3 models (70B, 8B) perform slightly below the flagship model but still exceed most other open-weight alternatives in their class.

3.5.2 Latency and Throughput

In the context of large language models, latency refers to the time taken for the system to produce an output after receiving a prompt, while throughput describes how many requests or tokens a model can process in each period. These two factors are central to evaluating the efficiency of inference in real-world applications.

Inference costs and latency for GPT-4 remain relatively high due to its Mixture-of-Experts architecture, which requires significant computational resources despite only activating two experts per forward pass. On average, GPT-4's inference latency is roughly three times greater than that of GPT-3's largest dense model.

LLaMA-3's 405B model, while dense and fully activated, also demands distributed inference infrastructure. Meta has implemented performance optimizations such as quantization to reduce latency. Nevertheless, the real-time responsiveness of LLaMA-3 remains limited without powerful hardware and efficient batching.

Smaller LLaMA-3 variants offer improved latency and can be deployed on more modest GPU clusters. For example, the 8B and 70B models can run on servers equipped with between 4 and 16 GPUs, achieving sub-second response times for shorter prompts.

3.5.3 Robustness and Safety

Both GPT-4 and LLaMA-3 demonstrate improvements in robustness and alignment with human preferences, though neither model is immune to hallucinations or biased outputs. OpenAI applied RLHF during GPT-4's fine-tuning process, improving safety and factual accuracy relative to GPT-3.5. GPT-4 performs well on benchmarks such as TruthfulQA, which assess a model's tendency to generate plausible but incorrect statements.
Meta has taken similar steps with LLaMA-3, including the development of Llama Guard, a safety layer designed to pre-screen and moderate model inputs and outputs. LLaMA-3 was also trained and fine-tuned on safety-aligned datasets. Despite these efforts, the open nature of LLaMA-3 means that responsibility for alignment and oversight falls to the users and community, whereas GPT-4's access remains tightly managed by OpenAI.

3.5.4 Multilingualism

GPT-4 exhibits strong multilingual performance. On translated versions of MMLU and other benchmarks, it consistently outperforms GPT-3 and GPT-3.5, even in low-resource languages like Welsh and Swahili. This reflects the breadth and diversity of its training corpus and its increased parameter count.
LLaMA-3, by design, includes extensive multilingual support. Approximately 8% of its training data is non-English, and its tokenizer has been explicitly augmented to include tokens for high-frequency terms in many languages. As a result, LLaMA-3 performs comparably to GPT-4 in multilingual evaluations, often closely tracking its results across a wide linguistic spectrum.

3.6 Strengths and Weaknesses

3.6.1 GPT-4 Strengths

GPT-4 has achieved strong performance across a variety of benchmarks, particularly on tasks requiring extensive knowledge and reasoning. Studies also report notable improvements in coding ability, multilingual understanding, and general problem-solving. Its multimodal extension allows it to process both text and images, expanding the range of potential applications. The model has been refined through reinforcement learning with human feedback, which contributes to improved accuracy and safer outputs. Today, GPT-4 underpins

many widely used products and services, demonstrating its practical reliability in real-world settings.

3.6.2 GPT-4 Weaknesses

Despite its strengths, GPT-4 remains a proprietary system. Users cannot inspect, modify, or self-host the model. Interaction is limited to API calls or usage through OpenAI's interfaces. Its inference costs are high, and deployment requires powerful infrastructure-at least 128 GPUs for effective operation. Moreover, although GPT-4 is significantly aligned, it still exhibits hallucinations, residual biases, and moderate context limits (~8K tokens in its base version, up to 32K or 128K in some enhanced variants).

3.6.3 LLaMA-3 Strengths

LLaMA-3 is a open and adaptable model. Meta has released multiple model sizes, allowing researchers and developers to fine-tune and deploy LLaMA-3 variants as needed. The 405B model displayed similar performances to GPT-4 in many contexts, and its 128K window works particularly well when it comes to handle long documents. The model's tokenizer and a training based on a multilingual dataset give it strong international applicability. Efficiency techniques such as GQA and FP8 quantization help balance performance and resource requirements. The community license fosters experimentation and innovation within defined boundaries.

3.6.4 LLaMA-3 Weaknesses

The scale of the 405B represent a limit for its real life applications as it requires specialized hardware together with a considerable amount of energetic resources. While LLaMA-3 includes alignment features like Llama Guard, it still lacks the centralized safety controls offered by GPT-4. Finally, while LLaMA-3's performance is strong, it almost never exceeds GPT-4’s capabilities, particularly when it comes to code generation. Like all current LLMs, it is still susceptible to hallucinations and embedded biases from training data.

Performance Benchmarks (Summary)

Benchmark (English)	**GPT-4**	**LLaMA-3 (405B)**	**GPT-3.5**	**GPT-4 (0-shot)**	**Notes**
MMLU (5-shot)	86.4%	~72–78%	70.7%	73.0%	General knowledge exams
GSM-8K (Math)	92.0%	80.9%	57.1%	79.9%	Grade school math
HellaSwag (Commonsense)	95.3%	~90% (est.)	85.5%	85.2%	Everyday reasoning
Code (HumanEval)	67%	~47%	48.1%	69.8%	Python coding
Multilingual MMLU	SOTA in 24/26	Comparable (Meta)	n/a	–	GPT-4 leads in many languages

Benchmark (English)	GPT-4	LLaMA-3 (405B)	GPT-3.5	GPT-4 (0-shot)	Notes
Safety (TruthfulQA)	Improved	Good (with Guard)	Lower	–	Resistance to adversarial falsehoods

These benchmarks confirm that while GPT-4 leads in most categories, LLaMA-3 follows closely and outperforms GPT-3.5 significantly. Smaller LLaMA models offer better latency and hardware flexibility, while GPT-4 remains more polished and reliable under controlled access.

3.7 Discussion

GPT-4 and LLaMA-3 represent two different approaches in the evolution of large language models. GPT-4 is a closed, highly engineered product, with a strong focus on performance and safety pushing on the bleeding edge of technological advancement offering features like multimodal integration, and safety alignment through centralized control. LLaMA-3, on the other hand, is an open, community-licensed platform designed for flexibility, transparency, and broad experimentation with the clear goal of providing a cheaper approach based on higher quality datasets.
Both models have made substantial progress over their predecessors, result of massive economic investments to cover substantial training and engineering costs.
For users in real life applications, the choice between GPT-4 and LLaMA-3 depends on context and constraints. GPT-4 offers superior performance out of the box but limits access and customizability. LLaMA-3, by contrast, allows researchers to fine-tune and host their own variants, making it ideal for customized solutions, provided the necessary compute resources are available.

Chapter 4: Cybersecurity in Health Coaching Applications

The increasing range of functional AI models in all their variations (LLMs, CNNs) has been accompanied by an equivalently large demand for applications employing such technologies. Health and wellness management applications functioning as virtual coaches now offer personalized support and feedback by integrating data collected from wearable devices such as smartwatches and wristbands. However, as the research pushes the boundaries of the available technology, it is critical to address a range of cybersecurity and privacy concerns stemming from such rapid advancements. As these devices and applications handle highly sensitive personal health data and operate over wireless channels (particularly Bluetooth), the design of robust security measures becomes not just important, but mandatory.

This chapter examines the cybersecurity issues relevant to cross-platform applications that connect smartphones with wearable devices over Bluetooth. It considers common vulnerabilities in such setups, explores current best practice and standards for Bluetooth security and looks at how authentication and pairing are handled. A specific focus is also reserved for regulatory requirements, in particular compliance with GDPR and HIPAA. Since the proposed application is expected to interact mainly with smartwatches and fitness bands through Bluetooth, the discussion emphasizes the specific security challenges that arise in this context.

4.1 Bluetooth Communication and Device Pairing

The fundamental mode of communication between wearables and smartphones is Bluetooth Low Energy (BLE), a widely adopted protocol designed for low-power, short-range connectivity. BLE represents a particularly suitable technology for continuous data transmission from fitness trackers and biometric sensors due to its low energy profile and ubiquity across mobile platforms.
Despite its convenience, BLE introduces multiple security concerns. Because BLE operates over unlicensed radio frequencies, it is considered vulnerable to **eavesdropping, spoofing, and man-in-the-middle (MITM)** attacks. These threats are magnified when default or insecure pairing modes are used, for instance, the "Just Works" pairing mode, an approach that does not require to verify the identity of the connecting device (often the default for devices without displays) provides no protection against MITM attacks.

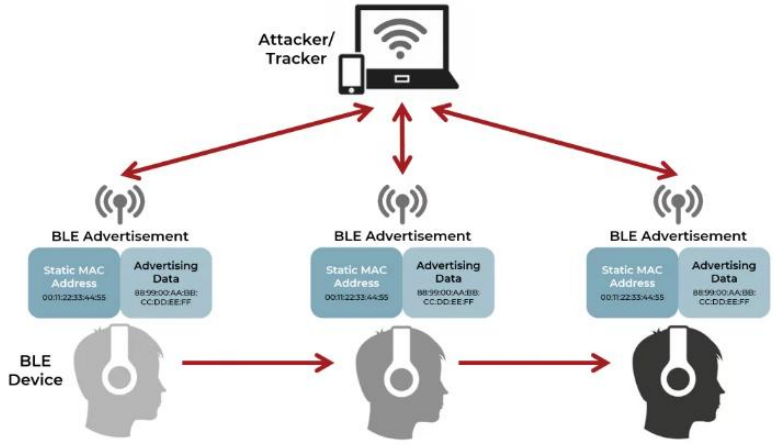

Fig.6 Layout of a potentially insecure network.

There exist more secure pairing protocols, such as Passkey Entry or Numeric Comparison, that by asking the user to match the input allows to verify his identity and protect against MITM attacks, unfortunately in situations involving simple wearables or waistbands that might lack touchscreen or inputs altogether such options may not be feasible. Thus, compensating mechanisms such as application-layer authentication, secure key exchange, or out-of-band verification become essential.
Once paired, devices usually establish a Long Term Key (LTK) using Elliptic Curve Diffie-Hellman (ECDH) and use it to encrypt ongoing communication using AES-128 encryption. Secure bonding ensures that future reconnections can use the previously negotiated key without re-pairing (reducing exposure), resulting in both better user experience and a higher level of security. Even this approach doesn't solve all the issues as the storing of the keys itself may represent a possible vulnerability, therefore apps must implement proper key-storage practices (such as using Android Keystore or iOS Keychain) to prevent exposure of cryptographic secrets.
Additionally, BLE provides a few privacy features, including MAC address randomization and Identity Resolving Keys (IRK) to help prevent user tracking through BLE advertising. The application should support and leverage these features to minimize the risk of passive surveillance.

4.2 Threat Models and Known Bluetooth Vulnerabilities

Bluetooth stacks have been historically prone to various classes of attacks. Some of the more common are:

1. Bluejacking and Bluesnarfing, which involve injecting unsolicited messages or extracting data from discoverable devices.
2. Bluebugging, which can exploit improperly secured devices to gain unauthorized access or control.
3. Bluesmacking, a form of Denial of Service (DoS) where oversized packets crash the Bluetooth stack.
4. MITM attacks, especially effective during the pairing phase in insecure modes.

Another critical set of vulnerabilities is collectively known as SweynTooth, potentially capable of affecting BLE chipsets used in many commercial wearable and medical devices by potentially allowing the attacker to trigger deadlocks, crashes or buffer overflow highlighting the importance of maintaining up-to-date firmware and applying security patches promptly. Counter measures against such threats generally consist in carefully managing BLE permissions, implementing device whitelisting, ensuring proper bonding protocols, and restricting pairing to user-initiated events. Additionally, BLE reliant applications should avoid operating in persistent discoverable mode, which can expose it to unsolicited pairing requests or data sniffing.

4.3 Authentication and Authorization

In a health ecosystem that involves multiple devices, authentication is required not only between the application and the wearables but also at the level of the individual user. While, as previously mentioned, Bluetooth offers protection at the link layer, an additional application-layer mechanism is needed to guarantee that data is received only from trusted sources. For wearables, this can be achieved by verifying unique device identifiers or by establishing a mutual cryptographic handshake during the initial pairing process. From a user perspective, secure login mechanisms (using OAuth 2.0, OpenID Connect, or biometric login) help ensure that only authorized individuals can access health data or settings.
The app should also maintain fine-grained authorization controls. For example, it should restrict access to certain data streams based on user roles or device trust levels. This is particularly relevant if the app scales to include healthcare providers or shared family access in future iterations.
All authentication tokens and session keys must be securely stored, rotated periodically, and transmitted only over encrypted channels. If expiration and signature validation are properly enforced JWT (JSON Web Tokens) can be used for stateless session management.

4.4 Data Privacy and Compliance (GDPR and HIPAA)

Due to the nature of the application gathering, processing, and storing personal health data is unavoidable these practices are regulated under the General Data Protection Regulation (GDPR) in the EU and the Health Insurance Portability and Accountability Act (HIPAA) in the United States.
Under GDPR, health data falls within the category "special category data," which requires explicit user consent for processing. Following one of the core principles of cybersecurity the regulation mandates privacy **by design** and **by default**, meaning data minimization, purpose limitation, and user control must be designed and considered from the earliest stages of development.

HIPAA, which applies to any app interacting with covered healthcare entities or storing identifiable health information in the U.S., imposes additional constraints. These include maintaining audit trails, role-based access control, and ensuring encrypted data transmission and storage.
To comply with these frameworks, the app should:

1. Collect only the data necessary for its intended function.
2. Provide transparent privacy policies and user consent dialogues.
3. Encrypt health data in transit (e.g., TLS 1.3 for network communication) and at rest (e.g., AES-256 for local storage).
4. Implement secure user authentication and automatic session expiration.
5. Ensure secure data deletion mechanisms in the event of account closure or withdrawal of consent.

A core aspect of these regulations is that failure to comply with them may not only result in legal penalties but also compromise the user's trust, an essential element for health-focused applications and development in general.

4.5 Secure Cross-Platform Development Practices

Given the need for wide adoption, the health coach app must support both Android and iOS platforms. The cross-platform framework selected, Flutter facilitates the development process as well as addressing the compatibility concerns but, at the same time, it also introduces unique security considerations.
For example:

1. Secure storage differences: sensitive data such as credentials must be stored using the platform-specific mechanisms provided by Android (Keystore) and iOS (Keychain). Flutter plugins must correctly map to these native services to avoid weaker storage on one platform.
2. Permission handling: runtime permissions for Bluetooth and health data access are managed differently across Android and iOS. The app should present consistent, transparent prompts to the user while respecting each system's security model.
3. Data validation across platforms: since Flutter bridges code to native APIs, all data exchanged between the app, peripherals, and system services should be validated on both Android and iOS to prevent inconsistencies or platform-specific vulnerabilities.
4. Session and key management: logout, device removal, or app uninstallation may trigger different cleanup processes on each OS. The application must ensure that cryptographic keys and session tokens are reliably cleared in all cases to prevent residual access.

Additionally, the BLE connection lifecycle should be controlled by the app itself, with connections actively initiated rather than passively accepted. Any disconnection event should trigger a state reset and, where appropriate, prompt the user for re-authentication. To further strengthen security, the app should enforce device whitelisting, periodic re-validation of trust, and consistent application of measures such as certificate pinning and code obfuscation across both platforms.

4.6 Update and Patch Management

An often-overlooked aspect of cybersecurity in mobile health apps is the ability to issue updates-both for the app itself and for any connected wearables. This is especially critical given the historical vulnerabilities in BLE stacks.

The app should include mechanisms for over-the-air (OTA) updates for its firmware and linked devices. In the absence of OTA support from the wearable manufacturer, the app should notify users when their device firmware is outdated or potentially vulnerable.
It is also recommended to incorporate root or jailbreak detection to limit application functionality on compromised devices, as these environments may allow unauthorized access to encrypted storage or network traffic.
Security logs should be maintained and, where applicable, sent to a backend monitoring service to detect anomalies such as repeated failed pairings, suspicious device identifiers, or unauthorized access attempts.

4.7 Summary and Recommendations

Security is not a one-time feature but a continuous process. In health applications, where the sensitivity of data and the risk to users is high, cybersecurity must be embedded throughout the software development lifecycle.
To ensure the safety and privacy of users, the proposed health coaching app must:

1. Use authenticated BLE pairing modes and avoid insecure defaults.
2. Encrypt all data in transit and at rest.
3. Follow platform best practices for secure data handling and storage.
4. Obtain and respect user consent as required by GDPR and HIPAA.
5. Implement secure authentication, session management, and fine-grained access controls.
6. Plan for regular updates and vulnerability mitigation.

By taking a proactive, privacy-by-design approach and aligning with industry standards, the application can not only protect its users but also build trust and position itself for regulatory approval and market success.

Chapter 5 Security Protocols for API Communication and Authentication

5.1 Introduction

The growth in popularity of distributed architectures and public networks calls for the protection of data in transit and the guarantee of identity authenticity as cornerstones of system design. This chapter examines four key technologies that supports secure API communication: **OAuth 2.0 with Proof Key for Code Exchange (PKCE)**, which manages user authorization; **Firebase Authentication**, which provides identity verification and user management; **JSON Web Tokens (JWTs)**, which enable the structured exchange and validation of claims to guarantee message integrity; and the **HTTPS/TLS protocol**, which encrypts data in transit to preserve confidentiality. Together, these technologies address all the layers of the trust problem and form a security architecture that is now fundamental to mobile and web applications.

5.2 OAuth 2.0 with Proof Key for Code Exchange (PKCE)

5.2.1 Background

OAuth 2.0 is the de facto authorization framework that allows third-party applications to access user resources without sharing user credentials. In the **Authorization Code Flow**, the client receives an authorization code from the authorization server, which it exchanges for an access token. While secure for confidential clients (e.g., server-side applications), this flow presents risks in **public clients** (e.g., mobile and single-page apps) because these cannot securely store a client secret. Attackers intercepting the authorization code could exchange it for tokens, impersonating the client.

5.2.2 PKCE Mechanism

To address this vulnerability, the **Proof Key for Code Exchange (PKCE)** extension introduces a dynamic secret. The client generates a high-entropy **code verifier** (e.g., 43–128 characters of random data). From this, it derives a **code challenge** using a transformation function (commonly SHA-256 followed by Base64URL encoding). During the authorization request, the client sends the code challenge to the server. When later exchanging the authorization code for tokens, the client must present the original code verifier. The server validates this by recomputing the code challenge and comparing it with the stored value. Only if they match is the access token issued. This ensures that intercepted authorization codes are useless without the corresponding verifier.

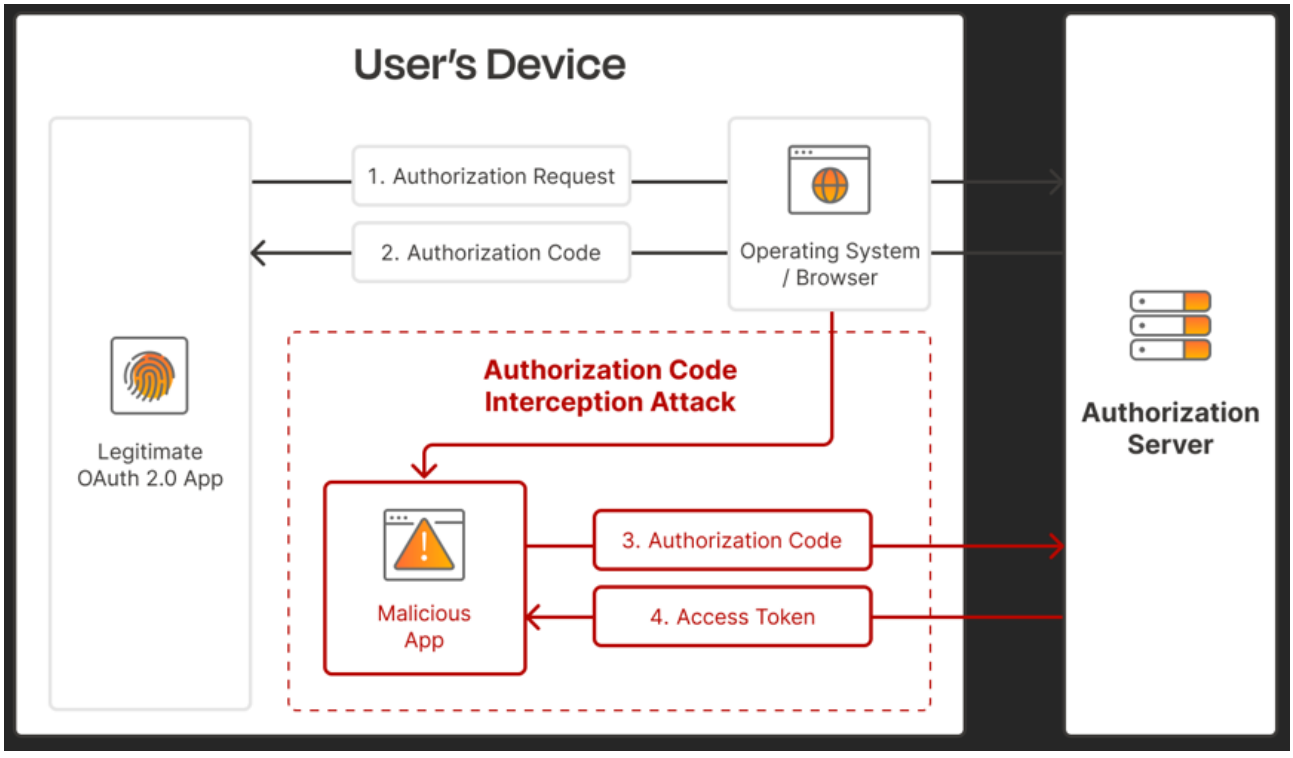

Fig.7 Representation of PKCE mechanism.

5.2.3 Flow

The PKCE-enhanced flow unfolds in the following steps:

1. **Code Verifier Creation** – The client generates a cryptographically random string.
2. **Code Challenge Derivation** – The challenge is derived via SHA-256.
3. **Authorization Request** – The client sends the authorization request with the code challenge.
4. **Authorization Grant** – The user authenticates, and the server returns an authorization code.
5. **Token Request** – The client submits the authorization code and the original code verifier.
6. **Token Issuance** – The server validates the verifier and issues tokens.

5.2.4 Security Properties

PKCE neutralizes interception attacks (code interception and replay). It also enhances security in environments where client secrets cannot be safely stored. Its adoption is recommended even for confidential clients to provide defense in depth.

5.3 Firebase Authentication Architecture

5.3.1 Overview

Firebase Authentication is Google's identity management solution, designed to simplify secure sign-in across platforms. It supports multiple authentication providers, including email/password, phone number, federated identity providers (Google, Facebook, Apple), and custom systems. It abstracts the complexity of credential management, token issuance, and secure session handling into a unified framework.

5.3.2 Client-Side Flow

When a user initiates authentication (e.g., with Google sign-in), the Firebase SDK manages the interaction with the identity provider, retrieves credentials, and exchanges them for a **Firebase ID token** and a **refresh token**. The ID token is a JWT containing the user's identity claims, signed by Google.

5.3.3 Backend Verification

On the backend, servers verify the ID token using Firebase Admin SDKs or by manually validating the token's signature and claims against Google's public keys. This guarantees that tokens were legitimately issued and have not been tampered with. Tokens include expiration times, ensuring that stale credentials cannot be reused.

5.3.4 Architecture Layers

The architecture can be understood in three tiers:

1. **Client Application** – Handles user input and invokes Firebase SDKs.

2. **Firebase Authentication Service** – Interfaces with identity providers, manages tokens, enforces authentication flows.
3. **Application Backend** – Validates tokens, associates them with application-specific sessions, and enforces authorization rules.

5.3.5 Advantages

This architecture reduces the complexity for developers while maintaining strong security guarantees, including multi-factor authentication, standardized token formats, and integration with Firebase's real-time database and cloud functions.

5.4 JSON Web Token Structure and Validation

5.4.1 Structure

A **token** is a compact piece of data issued by an authentication system and presented by a client to prove its identity or authorization status. Tokens allow services to verify requests without repeatedly asking for usernames and passwords, thereby improving both security and efficiency.

A **JSON Web Token (JWT)** is compact and URL-safe, it encodes claims as JSON objects. Each JWT consists of three parts separated by dots (.):

1. **Header**: specifies the algorithm used for signing (e.g., HS256, RS256) and the token type.
2. **Payload**: carries the claims, which may include registered fields such as iss (issuer), exp (expiration), aud (audience), and sub (subject), along with public or private claims defined by the application.
3. **Signature**: produced by signing the header and payload with a secret or private key, ensuring that the token has not been tampered with.

The resulting JWT has the form:
header.payload.signature

5.4.2 Validation

Validating a JSON Web Token (JWT) involves 5 checks to ensure its authenticity and correct usage:

- **Signature verification:** confirms that the token has not been modified since it was issued.
- **Expiration (exp):** prevents tokens from being accepted once expired.
- **Issuer (iss):** verifies that the token was created by the expected authority.

- **Audience (aud):** ensures the token is intended for the service or application that receiving it.
- **Not Before (nbf):** restricts the token from being used before a specified start time.

If any of these conditions fail, the token must be rejected as invalid.

5.4.3 Security Considerations

JWTs must be transmitted over secure channels (HTTPS/TLS) to prevent interception. Token with a lifetime longer than a certain amount should be avoided, periodically refreshing them guarantees to always access shorter lived and thus safer ones. Additionally, the algorithms involved should always be explicitly specified and of a secure nature (avoiding misconfiguration vulnerabilities such as the “alg=none” attack).

5.5 HTTPS/TLS for Secure API Communication

5.5.1 Overview

Transport Layer Security (TLS), often combined with HTTP (HTTPS), is the foundational protocol for securing communication over networks.

5.5.2 The TLS Handshake

The TLS handshake establishes a secure session between client and server:

1. **Client Hello**: The client proposes supported cipher suites and a random nonce.
2. **Server Hello**: The server selects cryptographic parameters and sends its digital certificate.
3. **Certificate Verification**: The client verifies the server’s certificate against trusted Certificate Authorities (CAs).
4. **Key Exchange**: Using asymmetric cryptography (RSA or Diffie-Hellman), both parties establish a shared session key.
5. **Session Key Confirmation**: Both client and server switch to symmetric encryption using the derived session key.

5.5.3 Security Properties

TLS ensures:

1. **Confidentiality**: all exchanged data is encrypted using symmetric session keys, which are negotiated through asymmetric cryptography during the handshake. This design combines the efficiency of symmetric encryption with the robustness of public-key methods.
2. **Integrity**: each record transmitted over a TLS connection includes a Message Authentication Code (MAC) or an AEAD (Authenticated Encryption with Associated

Data) tag. These mechanisms allow the receiver to detect even single-bit modifications, preventing tampering.

3. **Authentication**: servers present X.509 digital certificates issued by certified authorities. This process assures the client that it is communicating with the legitimate endpoint.
4. **Forward Secrecy**: modern TLS versions (1.2 with EDH, and by default in TLS 1.3) establish session keys through ephemeral key exchanges. This guarantees the safety of previously exchanged message even after a long-term private key is compromised

5.5.4 Relevance to APIs

In API communication, TLS is indispensable. It guarantees that sensitive data (e.g., tokens, credentials, personal data) is not exposed to attackers, while also preventing man-in-the-middle attacks. Combined with strong authentication protocols, TLS completes the security model by safeguarding the transport layer.

5.6 Conclusion

In summary, securing a multi-platform application requires combining multiple techniques adopted in the early development stages rather than relying on a single safeguard. Bluetooth protections, cross-platform secure storage, and session controls reduce the risk of device spoofing and unauthorized data access. At the application and network layers, OAuth 2.0 with PKCE, Firebase Authentication, JWT validation, and HTTPS/TLS work together to prevent token forgery, code interception, and data exposure. Their combined use creates a layered defense providing resiliency against both technical exploits and privacy breaches.

6. Health Lesson Generator: A Firebase-Integrated Web Application for Educational Content Management

6.1 introduction

The Health Lesson Generator is a web tool developed to support the admin of the mobile application in creating and managing the education aspect of the project. The application allows to create and manage dynamic and scalable content to be leveraged in the mobile app. Built on Flask, a lightweight Python web framework, the system integrates cloud-based database solutions with traditional web development practices.

The primary objective of this tool is to provide the admin with an intuitive interface for developing structured health lessons and related quizzes in a format readily available for use on the mobile app through standardized API endpoints. This dual-purpose architecture eliminates data silos and ensures consistency across multiple platforms.

6.2 System Architecture

6.2.1 Application Framework

The Health Lesson Generator is built upon Flask 2.3.3, chosen for its simplicity, flexibility, and rapid development capabilities additionally its lightweight nature makes it even more suitable for this application, which requires quick iteration and deployment. The framework's modular design allows for clean separation of concerns between data management, business logic, and presentation layers.

The application follows the Model-View-Controller (MVC) architectural pattern, where:

1. **Models** are represented by Firebase Firestore documents
2. **Views** are implemented through Jinja2 templates with Bootstrap 5.3.0 styling
3. **Controllers** are Flask route handlers that manage HTTP requests and responses

6. 2.2 Database Architecture

The system utilizes Google Firebase Firestore, a NoSQL document database that provides real-time synchronization and automatic scaling. The database structure consists of two primary collections:

Lessons Collection: Each document contains structured lesson data including:

1. Metadata (title, category, subcategory, difficulty level).
2. Content structure (sections, tags, duration estimates).
3. Temporal information (creation and modification timestamps).
4. Categorical organization aligned with health domain taxonomy.
5. Language information.

Quizzes Collection: Assessment documents containing:

1. Question arrays with multiple-choice options.
2. Correct answer indices for automated grading.
3. Categorical alignment with corresponding lessons.
4. Assessment metadata for tracking and organization.
5. Language information.

This NoSQL approach provides flexibility in data structure evolution and supports the varied content types typical in educational materials.

6.2.3 Authentication and Security

The application implements Firebase Admin SDK authentication using service account credentials. This approach provides:

- Secure server-to-server communication with Firestore

- Granular access control through Identity and Access Management (IAM)

- Encrypted data transmission through HTTPS protocols

- Credential management through JSON key files with proper access restrictions

All the aforementioned protocols are discussed in the 2 cybersecurity related chapters.

6.3. Functional Architecture

6. 3.1 Content Management Interface

The web interface provides comprehensive content management capabilities through a series of coordinated views:

Creation Workflows: Streamlined forms for lesson and quiz creation with:

1. Dynamic category selection with hierarchical subcategory population
2. Section-based content organization for improved readability
3. Tag-based content classification for enhanced discoverability
4. Automated metadata generation including duration estimation

Editorial Functions: Full CRUD (Create, Read, Update, Delete) operations with:

1. Real-time content preview capabilities
2. Version control through timestamp tracking
3. Bulk operations for efficient content management
4. Content validation to ensure data integrity

Organizational Systems: Taxonomical content organization featuring:

- Five primary health categories (General Health, Sleep & Recovery, Nutrition & Diet, Physical Activity, Mental Wellness)

- Hierarchical subcategory structures for granular classification

- Tag-based cross-referencing for content discoverability

- Difficulty progression systems for educational scaffolding

6.3.2 API Architecture

The RESTful API design follows standard HTTP conventions and provides comprehensive data access:

Endpoint Structure:

1. `GET /api/lessons` - Retrieves all lesson documents
2. `GET /api/lesson/<id>` - Fetches individual lesson by unique identifier
3. `GET /api/quizzes` - Returns complete quiz collection
4. `GET /api/quiz/<id>` - Provides specific quiz document
5. `GET /api/subcategories/<category>` - Dynamic subcategory enumeration

Data Serialization: All responses utilize JSON serialization with consistent schema:

```json
{
 "id": "unique_identifier",
 "title": "content_title",
 "sections": ["array_of_content_sections"],
 "tags": ["categorization_tags"],
 "metadata": {
  "category": "primary_classification",
  "difficulty": "learning_level",
  "duration": "estimated_completion_time"
 }
}
```

6.3.3 Export Functionality

The system provides multiple data export mechanisms:

Individual Exports: JSON file downloads for specific lessons or quizzes, enabling content portability and backup functionality.

Bulk Exports: Comprehensive data dumps including all lessons, quizzes, and categorical metadata, facilitating system migration and data analysis.

Format Standardization: All exports maintain consistent JSON schema, ensuring compatibility across different consumption platforms.

6. 4 Technical Implementation

6. 4.1 Database Abstraction Layer

The `firebase_config.py` module implements a comprehensive database abstraction layer that:

1. Encapsulates all Firestore operations within dedicated classes

2. Provides error handling and fallback mechanisms
3. Implements connection pooling and resource management
4. Offers both synchronous and asynchronous operation support

6. 4.2 Error Handling and Resilience

The application implements multiple layers of error handling:

1. **Connection Fallback**: Automatic fallback to mock data during Firebase connectivity issues
2. **Transaction Management**: Atomic operations for data consistency
3. **Input Validation**: Server-side validation for all content creation and modification operations
4. **Graceful Degradation**: Continued operation even with partial system failures

6.4.3 Performance Optimizations

Several optimization strategies enhance system performance:

1. **Lazy Loading**: Content sections loaded on demand to reduce initial page load times
2. **Caching**: Strategic caching of frequently accessed content and metadata
3. **Pagination**: Large content sets delivered through paginated responses
4. **Compression**: JSON response compression for reduced bandwidth usage

6. 5 Integration Capabilities

6.5.1 Mobile Application Integration

The API-first design facilitates seamless integration with mobile applications through:

1. **Cross-Platform Compatibility**: JSON APIs consumable by iOS, Android, and cross-platform frameworks
2. **Real-Time Synchronization**: Firebase's real-time capabilities ensure content consistency across platforms
3. **Offline Capabilities**: Support for offline content caching and synchronization

6.5.2 Extensibility Framework

The modular architecture supports future enhancements:

- **Plugin Architecture**: Modular design allows for feature extensions without core system modifications

- **API Versioning**: Structured API versioning supports backward compatibility during system evolution

- **Content Type Extensions**: Framework supports additional content types beyond lessons and quizzes

6.6 Conclusion

The Health Lesson Generator represents a successful integration of modern web development practices with cloud-based database solutions. The system's architecture demonstrates effective separation of concerns, scalable data management, and comprehensive API design. The Firebase integration provides robust data persistence and real-time synchronization capabilities, while the Flask framework ensures rapid development and deployment.

The tool's dual-purpose design successfully addresses the needs of both content creators and mobile developers, providing a unified platform for health education content management. The JSON-first architecture ensures data portability and cross-platform compatibility, while the hierarchical content organization supports educational best practices.

Future developments could include advanced analytics capabilities, **multi-language support**, and enhanced collaboration features. The modular architecture and comprehensive API design provide a solid foundation for these potential enhancements, ensuring the system's continued relevance in the evolving landscape of educational technology.

Bibliography

[1] Okta Developer, "What is Proof Key for Code Exchange (PKCE)?," 2024. [Online]. Available: https://developer.okta.com/docs/guides/implement-grant-type/authcodepkce/main/

[2] Sysco, "Getting Started with Firebase Authentication," 2023. [Online]. Available: https://sysco.no/getting-started-firebase-authentication/

[3] jwt.io, "Introduction to JSON Web Tokens," 2023. [Online]. Available: https://jwt.io/introduction

[4] Cryptr, "Understanding JWT Validation," 2024. [Online]. Available: https://cryptr.co/blog/understanding-jwt-validation

[5] Cloudflare, "What happens in a TLS handshake?," 2023. [Online]. Available: https://www.cloudflare.com/learning/ssl/what-happens-in-a-tls-handshake/

[6] Cloudflare, "What is HTTPS?," 2023. [Online]. Available:

https://www.cloudflare.com/learning/ssl/what-is-https/
[7] P. Cunningham, M. Cord, and S. J. Delany, “Supervised learning,” in *Machine Learning Techniques for Multimedia*. Springer, 2016, pp. 21–49.
[8] P. C. Wong, “Unsupervised machine learning,” in *Computational Psychometrics: New Methodologies for a New Generation of Digital Learning and Assessment*. Springer, 2021, pp. 173–193.
[9] R. Yamashita, M. Nishio, R. K. Do, and K. Togashi, “Convolutional neural networks: an overview and application in radiology,” *Insights into Imaging*, vol. 9, pp. 611–629, 2018.
[10] A. Vaswani et al., “Attention is all you need,” in *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, 2017, pp. 5998–6008.
[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” in *Proc. NeurIPS*, 2012, pp. 1097–1105.
[12] T. B. Brown et al., “Language models are few-shot learners,” in *Proc. NeurIPS*, 2020, pp. 1877–1901.
[13] T. B. Lee and S. Trott, “Large language models, explained with a minimum of math and jargon,” *Understanding AI*, Jul. 27, 2023. [Online]. Available: https://www.understandingai.org/p/large-language-models-explained-with
[14] N. Comly, M. Lee, and W. Locke, “Predicting change in BMI using MyFitnessPal,” Stanford Univ., CS229 Project, Dec. 2019. [Online]. Available: https://cs229.stanford.edu/proj2019aut/data/assignment_308875_raw/26477813.pdf
[15] K. B. Johnson et al., “Precision medicine, AI, and the future of personalized health care,” *Clinical and Translational Science*, vol. 14, no. 1, pp. 86–93, 2020. [Online]. Available: https://doi.org/10.1111/cts.12884
[16] Choplife, “Smarter Eating, Smarter Fitness: AI supports your goals,” *Technology and Operations Management MBA Student Perspectives, Harvard Business School Digital Initiative*, Nov. 13, 2018. [Online]. Available: https://d3.harvard.edu/platform-rctom/submission/smarter-eating-smarter-fitness-ai-supports-your-goals/
[17] MyFitnessPal, “Meal Scan FAQ,” 2020. [Online]. Available: https://support.myfitnesspal.com/hc/en-us/articles/360045761612-Meal-Scan-FAQ
[18] Lark Health, “AI and digital health resources,” 2023. [Online]. Available: https://www.lark.com/resources/lark-health-ai-artificial-intelligence
[19] Freeletics, “AI and your Coach,” 2023. [Online]. Available: https://www.freeletics.com/en/blog/posts/AI-and-your-Coach/
[20] “How AI is transforming fitness apps,” *Health and Fitness Association*, 2023. [Online]. Available: https://www.healthandfitness.org/improve-your-club/how-ai-is-transforming-fitness-apps/
[21] OpenAI, “GPT-4 Technical Report,” arXiv preprint, Mar. 2023. [Online]. Available: https://arxiv.org/abs/2403.03346
[22] M. Touvron et al., “LLaMA: Open and Efficient Foundation Language Models,” arXiv preprint, Feb. 2023.
[23] H. Touvron et al., “LLaMA 2: Open Foundation and Fine-Tuned Chat Models,” arXiv preprint, Jul. 2023.

[24] A. Grattafiori et al., "The LLaMA 3 Herd of Models," arXiv preprint, Jul. 2024.
[25] "GPT-4 Architecture, Datasets, Costs and More (Leaks)," *The Decoder*, Jul. 2023.
[26] OpenAI, "GPT-4," OpenAI.com, 2024. [Online]. Available: https://openai.com/research/gpt-4
[27] Meta AI, "Introducing Meta LLaMA 3," Meta AI Blog, Apr. 18, 2024. [Online]. Available: https://ai.meta.com/blog/introducing-meta-llama-3/
[28] Olga, "LLaMA 3 License Explained," *DEV Community*, Apr. 19, 2024. [Online]. Available: https://dev.to/olga/llama3-license-explained
[29] Meta AI, "LLaMA 2 Community License Agreement," Apr. 2023. [Online]. Available: https://ai.meta.com/llama/license
[30] A. OpenAI, "GPT-4 (Azure) Transparency Report," Stanford CRFM, 2023.
[31] Bluetooth SIG, "Bluetooth Core Specification v5.4," Feb. 2023. [Online]. Available: https://www.bluetooth.com/specifications/specs/core-specification/
[32] A. Levy, "Bluetooth Low Energy (BLE) Security Overview," Nordic Semiconductor, 2022. [Online]. Available: https://www.nordicsemi.com/
[33] U.S. Food and Drug Administration, "Cybersecurity in Medical Devices: Quality System Considerations and Content of Premarket Submissions," Sep. 2023.
[34] European Commission, "General Data Protection Regulation (GDPR)," Regulation (EU) 2016/679, 2016. [Online]. Available: https://gdpr.eu/
[35] U.S. Department of Health and Human Services, "Health Insurance Portability and Accountability Act of 1996 (HIPAA)," 45 CFR Parts 160, 162, 164, 1996. [Online]. Available: https://www.hhs.gov/hipaa/
[36] J. Padgette, K. Scarfone, and T. Grance, "Guide to Bluetooth Security," NIST Special Publication 800-121 Rev. 2, Sep. 2017.
[37] OWASP, "Mobile Security Testing Guide," OWASP, 2023. [Online]. Available: https://owasp.org/www-project-mobile-security-testing-guide/
[38] M. Ryan, "Bluetooth: With Low Energy Comes Low Security," in *Proc. 7th USENIX Workshop on Offensive Technologies (WOOT)*, Washington, D.C., 2013.
[39] E. Rios, "SweynTooth: Unleashing Mayhem over Bluetooth Low Energy," SUTD, 2020. [Online]. Available: https://asset-group.github.io/disclosures/sweyntooth/
[40] Google, "Bluetooth Overview | Android Developers," 2024. [Online]. Available: https://developer.android.com/guide/topics/connectivity/bluetooth
[41] Apple Inc., "Core Bluetooth | Apple Developer Documentation," 2024. [Online]. Available: https://developer.apple.com/documentation/corebluetooth
[42] J. Logan, "Bluetooth LE Security and Privacy in Wireless Audio Devices," *Cardinal Peak Blog*, Aug. 10, 2023. [Online]. Available: https://www.cardinalpeak.com/blog/bluetooth-le-security-and-privacy-in-wireless-audio-devices. [Accessed: Sep. 11, 2025].