Plag cloud

by sharyar khan

Submission date: 14-Apr-2024 12:51PM (UTC+0300)

Submission ID: 2348976931

File name: Cloud_Cost-optimization_using_Resource_Recommendation_System.docx (1.93M)

Word count: 7263

Character count: 45234

CLOUD COST-OPTIMIZATION USING RESOURCE RECOMMENDATION SYSTEM

Kalaiselvi R[1] Thiruvaazhi U[2] Dharshan S[3] Kavya R[4] Priyadharshini S[5]

[1]*Department of Information Science and Engineering, Kumaraguru College of Technology, India*
E-mail: kalaiselvi.r.ise@kct.ac.in
[2]*Department of Information Science and Engineering, Kumaraguru College of Technology, India*
E-mail: thiruvaazhi.u.ise@kct.ac.in
[3]*Department of Information Science and Engineering, Kumaraguru College of Technology, India*
E-mail: dharshan.20is@kct.ac.in
[4]*Department of Information Science and Engineering, Kumaraguru College of Technology, India*
E-mail: kavya.20is@kct.ac.in
[5]*Department of Information Science and Engineering, Kumaraguru College of Technology, India*
E-mail: priyadharshini.20is@kct.ac.in

Abstract
Cloud computing has emerged as an indispensable component of contemporary enterprise operations delivering a diverse array of services spanning multiple sectors however the inherent complexity of the cloud ecosystem presents significant challenges for businesses particularly novices who encounter difficulties in selecting appropriate services and managing expenses existing recommendation systems often overlook the needs of beginners assuming a certain level of expertise this paper proposes a novel approach to address these challenges through rapid engineering leveraging large language models Large Language Models(LLMs). This methodology gathers comprehensive customer requirements recommends suitable cloud services estimates costs and provides tailored integration guidelines by prioritizing expedient engineering this approach harnesses the capabilities of LLMs while incorporating human expertise facilitating informed decision-making regarding cloud adoption and management this research underscores the critical importance of leveraging advanced ai methodologies in tandem with human insights to navigate the complexities of the cloud environment effectively.

Keywords:
Large Language Model, Artificial Intelligence, Extensible Markup Language, New cloud User, Recommendation System, Cost-optimization, Generative pre-trained transformers, Gemini, Cloud Services.

1. INTRODUCTION

1.1. INTRODUCTION TO THE ARTICLE

In recent years, the rapid expansion of cloud services from various providers has delivered significant benefits to consumers. However, selecting the most appropriate cloud services from a wide array of options poses a challenge for new users. Manual approaches have proven inefficient in this regard. Consequently, the adoption of automated recommendation systems emerges as a promising solution to address this issue. These systems adeptly provide tailored service recommendations to cloud users based on their specific needs. This article proposes a cloud service recommendation approach based on user requirements using Large Language Models (LLMs). The primary objective is to develop and implement a recommendation system for optimizing cloud resources. This system aims to assist users in making informed decisions by offering personalized, data-driven recommendations for selecting, managing, and optimizing cloud resources. LLMs play a crucial role in this process, utilizing advanced natural language processing and pattern recognition techniques to analyze user input. They interpret user needs expressed in various forms, identify relevant features and constraints, and match them with available cloud services. By leveraging LLMs, the recommendation system can recommend the most suitable cloud solutions tailored to each user's requirements. This article seeks to enhance the user experience, facilitate cost savings, and improve the efficiency of cloud resource allocation. Ultimately, users are empowered to make well-informed decisions in their cloud service selection process.

1.2. SCOPE OF THE ARTICLE

The aim is to develop recommendation system in the context of cloud computing to offer insights and guidance for the optimization of cloud services.

- The task involves gathering essential information from the user, their specific needs, budget constraints, and technological proficiency.

- A comprehensive contextual analysis will be conducted to understand the user's environment and objectives. This analysis will cover the use-case identification.

- Incorporating a Large Language Model to process and interpret the user input may be significant. The LLM will play a crucial role in understanding natural language descriptions of user requirements.

- To ensure the effectiveness of the recommendation system, and establishing clear evaluation metrics such as accuracy, precision, recall, cost estimation.

2. LITERATURE SURVEY

As cloud records its footprint in many domains like healthcare, it is required to make its usage very friendly [1].

Dehua Kong et al stated the challenge of helping cloud service users to select the most suitable services from a plethora of options. Authors conduct experiments comparing TRSC with traditional collaborative filtering and random recommendation, demonstrating that TRSC provides higher accuracy. The system proposed by Dehua Kong et al has a potential to assist users in making informed cloud service choices in a rapidly evolving cloud market, but requires enhancement in system's quality pertinent to resource suggestions [2].

Chao Luo et al outlined a recommendation system for cloud products based on real-time cost data and a knowledge graph. A knowledge modeling process was proposed, involving the gathering of information from various sources, including cloud vendors' official websites, forums, and blogs. The crawler is responsible for collecting these data according to the data collection strategy, ensuring the retrieval of the most up-to-date insights or expertise. Detailed product attributes, including characteristics, features, and related details, are represented in dataset T1, a mixed set comprising information on various pall productions. Dataset T2, a subset of T1, is focused on the sub-product "ECS" from Elastic Compute, providing granular details such as attributes and features. T2 is filtered through targeted queries and constraints related to the manufacturer to extract specific ECS sub-product information. "IS ON" relationships are derived from both T1 and T2. Information about related products suitable for building a use case is provided in a part of the T1 dataset. Another part of T1 maps to query field values in the relational database (MySQL), specifically ECS codes serving as identifiers like "product_class_id." These codes are used to look up product attributes obtained from T2. By leveraging these codes and product attribute data from T2, product knowledge can be presented, and the full portfolio of a manufacturer's offerings can be retrieved for a given query. Final recommendations are generated using data from both T1 and T2 [3].

Rui-dong Qi et al has analyzed a process to enhance cloud service recommendations with improved scalability. An elastic service recommender process (ERP) was proposed and it is capable of constructing numerous recommender strategies in short time period to generate real-time recommendations. This is sent to user and based on the feedback the recommender algorithm is adjusted and it restarts the recommendation. Cosine similarity is used to measure the similarity between users preferences and it is clustered using Density-based clustering algorithm. The ERP outperforms other baseline algorithms in terms of accuracy, with improvements of 18% and 10% in MAE and RMSE values, respectively [4].

Abid Mahmood et al stated methodology for selecting cloud services based on intelligent agent systems, aimed at addressing the challenge of choosing the most suitable cloud services from a large number of options. A cloud service selector (CSS) was designed with two modules, including a graphical user interface and a multiple agent system. It was concluded by the author that MAS is practically applicable for solving the problems of cloud service vendor selection in multidimensional techniques [5].

Rahma Djiroun et al identified a new approach for cloud service recommendation, combining learning techniques and data mining. It involves analyzing service content and user behavior, with a pruning process to refine recommendations. Two evaluation measures, M1 and M2, calculated at M1=0.37 and M2=0.35 respectively, indicate a good system relevance, but these results are non-persuasive due to being obtained from simulated user interactions[6].

Wenqi Fan et al stress the significance of Recommender Systems (RecSys) in improving online user experiences and managing information overload. They highlight the emergence of Large Language Models (LLMs), emphasizing their robust language understanding and generation capabilities. LLMs can adapt to new tasks and domains, making them promising for enhancing recommender systems. The authors introduce methods for converting recommendation data into natural language and discuss techniques like In-context Learning (ICL) and instruction tuning for customizing LLMs in recommendation tasks. [7].

Likang Wu et al stated that system focused on predicting the changing conditions of users, based on their evaluations and service conditions. This system aimed to forecast the quality of service (QoS) conditions for users by analyzing their feedback and the quality of service values. It involved calculating the QoS demand vector for each user. The recommendation algorithm comprised two steps: Neighbors and Services Screening. The system calculated the similarity between the target user and others in the same community, identifying the most similar users as neighbors and considering the services they had interacted with. Anticipated QoS values for these services were computed based on user feedback, average service conditions of the target user, and QoS values from service providers. The system estimated the suitability of each service for the target user, considering the feedback from neighboring users, their similarity, and differences in QoS values. Top-rated services were recommended to the target user. Thus, the system was found to enhance recommendation precision without compromising effectiveness.[8].

An Zhang et al have developed "Agent4Rec," a movie recommendation simulator. Central to their approach are generative agents, built on LLMs (Large Language Models). These agents consist of three key modules: a profile, memory and action. The memory module is inspired by human cognitive processes, enabling agents to store and recall past interactions and emotions to generate coherent behaviors. The action module offers a variety of actions, including those influenced by user preferences and emotions, to simulate realistic user behaviors. Building the recommendation environment within Agent4Rec is crucial, involving simulating interactions between generative agents and the recommendation environment [9]

Nidhi Hegde et al has stated the utilization of Large Language Models (LLMs) in tasks related to answering questions about image documents. They approached the task by considering it solely as an LLM task, without incorporating a vision encoder (image-to-text model). Several crucial factors impacting the effectiveness of the LLM-only approach were thoroughly examined. This approach obtained from this research shed light on the importance of reading order, with appropriate heuristics significantly enhancing the performance of the LLM-only model. Author concluded that the findings can provide guidance for researchers when choosing datasets for their investigations, particularly in scenarios where a vision encoder may not be deemed essential [10].

Liangmin Guo et al. propose a method tailored for cloud environments to predict users' changing requirements by analyzing their service evaluations and ratings. By examining historical data, the system generates a QoS requirement vector for each user and identifies similar users as neighbors. Predicted QoS values are computed for services based on community evaluations and user ratings. A matching degree is calculated to assess service suitability, and ratings are predicted for each service using neighboring users' feedback. Finally, top-rated services are recommended to the target user, enhancing recommendation accuracy while maintaining efficiency.[11]

Tao Fan et al introduced a framework called FATE-LLM, is an industrial-grade solution designed for large language models (LLMs).Two primary challenges faced by LLMs in real-world applications are highlighted: the limitation of adoption by smaller enterprises due to the high computing resources required for training, and the necessity for large amounts of high-quality data scattered among different entities. FATE-LLM is intended to address these challenges by providing a framework that enables federated learning for large language models, promoting efficient training methods, protecting intellectual property, and ensuring data privacy during both training and inference. The components: the Communication-Efficient Hub, the FedLLM Model Hub, and the FedLLM Privacy Hub. It supports various scenarios, including federated homogeneous and heterogeneous LLMs, co-tuning LLMs, and offsite tuning. Emphasis is placed on the importance of privacy-preserving mechanisms, such as federated intellectual property protection and secure aggregation, in ensuring data privacy during federated learning. Experiments are conducted on a scenario where multiple clients collaboratively fine-tune their ChatGLM-6B models through federated learning, demonstrating the effectiveness of the proposed framework [12].

Wenxuan Zhang et al. presented a strategy to close the knowledge gap between generative big language models (BDLM) and domain-specific models in personalised recommendation. Personalised recommendation systems may be improved with BDLM, as it bridges the knowledge gap between large language models (LLMs) and domain-specific models. To take use of the advantages of both models, a collaborative training programme and an information exchange module are used. LLMs add general knowledge and reasoning skills, and domain-specific models improve suggestions by providing details about community behaviour patterns. Experimental findings on actual datasets show that BDLM is successful in enhancing recommendation performance when compared to state-of-the-art approaches. The method has potential for optimising recommendations in situations involving both domain-specific models and LLMs [13].

Junjie Zhang et al introduced a novel approach to developing recommender systems by leveraging large language models (LLMs) for instruction following. Unlike traditional methods that rely on historical behavior data, the proposed method considers user preferences and needs expressed in natural language instructions. The authors design a specific instruction format encompassing preference, intention, and task form, and create 39 instruction templates to generate 252K user-personalized instructions. The open-source LLM (3B Flan-T5-XL) is then "instruction-tuned" to adapt to recommender systems. Experimental results demonstrate the approach's effectiveness, outperforming competitive baselines, including GPT-3.5, on various tasks. The proposed method emphasizes user-friendly interaction, allowing users to communicate their needs in natural language for more accurate recommendations. The paper concludes by highlighting the potential for scaling LLMs, extending context length, and applying the approach in multi-turn interaction scenarios in future work [14].

Miranda Zhang et al looked at how cloud computing has significantly affected internet-based application services, emphasising the wide range of options accessible to decision-makers. Based on the Analytics Hierarchy Process (AHP) they presented a real-time Quality of Service (QoS) aware multi-criteria decision-making approach designed specifically for choosing Infrastructure as a Service (IaaS) cloud services. This technique makes it easier to identify the most effective cloud service configuration at the Infrastructure as a Service layer by allowing customers to define both design-time and real-time QoS requirements. With a focus on quality of service, the system seeks to streamline the process of choosing and evaluating IaaS solutions while offering cloud providers useful market intelligence. Subsequent efforts will involve the integration of Service Level Agreements (SLAs) with legal compliance, the improvement of data collection techniques, and the execution of experiments using real-time network QoS data under unpredictable conditions such as network outages and congestion. [15].

Lanning Wei et al stated a novel approach, named Auto2Graph, which makes use of Large Language Models (LLMs). Tasks are divided into three phases using this method: determining the learning intent, using AutoGraph to configure solutions, and producing responses. Important operations including the analysis of data, ML setup, architecture searching, and hyper-parameter changing are handled by AutoGraph agents. Auto2Graph exhibits human-like decision-making by agents and is effective across a range of datasets and tasks. Graph Neural Networks (GNNs) are a popular approach for solving customised problems using graph-structured data in domains like as chemistry, social networks, and e-commerce. Nonetheless, handling heterogeneous data continues to be difficult, requiring an effective strategy. The system's ability to make cloud computing more accessible to regular consumers is emphasised. It makes multi-criteria Infrastructure as a Service (IaaS) offer selection easier while taking Quality of Service (QoS) into account and provides cloud providers with market insights. [16].

Daixing Zhong et al have suggested a solution system for recommendations for cloud-based environments with the goal of helping consumers choose the best services from different cloud providers. The solution tackles issues including limited data, scaling, and the cold-start problem that come with conventional distributed filtering recommendation algorithms. To tackle uncertainty, an approximate multi-dimensional matrices model that combines decision timings and decision tags is used in conjunction with a collaborative filtering method that incorporates a rough decision mechanism. This cooperative filtering technique takes into account both user (decision times) and item (decision tag) factors to maximise recommendation accuracy in cloud-based environments The methodology involves building a user-item scores matrix, introducing decision tags and decision times, and addressing sparsity issues in the dataset. The study contributes to the field of recommender systems, providing insights into enhancing recommendation accuracy in cloud-based environments and addressing specific challenges related to data sparsity and scalability [17].

Xu Huang et al has outlined a framework called InteRecAgent that aims to combine the strengths of large language models (LLMs) and recommender models. The frame includes three crucial factors: a participated memory machine for effective communication between tools, allowing intermediate countries to be stored, enabling effective communication between different tools in the system. Next, a dynamic demonstration-stoked plan-first prosecution strategy for task planning. And a reflection strategy where another LLM acts as a critic to estimate the quality of results and address crimes during prosecution. Experimental results on three public datasets (Steam, MovieLens, Amazon Beauty) are presented in the paper, demonstrating the effectiveness of InteRecAgent. The frame outperforms general-purpose LLMs in conversational recommender systems, particularly in disciplines that are less covered by general world knowledge. The proposed frame aims to homogenize conversational recommender systems by integrating LLMs with traditional recommendation models, furnishing a more natural and flawless stoner experience. [18].

Yunfan Gao et al has stated a novel recommender system paradigm called Chat-Rec, which combines Large Language Models (LLMs), like ChatGPT, with traditional recommender systems. The goal is to address challenges such as poor interactivity, explainability, and the cold-start problem in existing recommender systems.The proposed Chat-Rec framework enhances interactivity and explainability in recommendations. It enables multi-round interactions, allowing user preferences to be learned during conversations, leading to more personalized and context-aware recommendations. The experiments conducted on the MovieLens 100K dataset show that Chat-Rec outperforms traditional recommender systems in terms of recommendation quality and rating prediction. Overall, Chat-Rec presents a promising technique for enhancing recommender systems by incorporating LLMs, improving user interaction, explainability, and extending the system's capabilities to handle new items and cross-domain recommendations [19].

Keqin Bao et al has designed a framework, TALLRec, to enhance the efficiency of LLMs in respective of recommendations. They found that when faced with recommendation problems, LLMs frequently perform poorly because there is a discrepancy between the tasks they were trained on and the real recommendation tasks, and because they were not given enough recommendation data in the initial training stage. They suggest using a two-stage tuning architecture to overcome these problems. The first step, known as alpaca tuning, makes use of the data that is already available to improve LLMs' capacity for generalisation. TALLRec surpasses conventional recommendation models and LLMs trained with Throughout-context Learning in their trials centred on book and movie recommendations. This emphasises how crucial it is to adjust in accordance with detailed instructions in order to get past earlier constraints. TALLRec has excellent domain-crossing flexibility and suggesting promising prospects for recommendation tasks even with limited data.[20]

Junling Liu et al has introduced LLMRec, a recommender system leveraging Large Language Models (LLMs) for benchmarking on various recommendation tasks. It reveal that while LLMs show moderate proficiency in accuracy-based tasks, their performance excels in explainability-based tasks. Additionally, it includes the impact of supervised fine-tuning (SFT) on LLMs to enhance instruction compliance. The results indicate that SFT significantly improves LLMs' capabilities, aligning them better with recommendation instructions, especially in tasks requiring formatted outputs. The study compares LLMs with baseline methods and emphasizes the effectiveness of LLMs, particularly ChatGPT, in generating clear and rational explanations for recommendation results [21].

Shivanshu Shekhar et al has stated the importance of token optimization in natural language processing tasks is emphasized, with a focus on heuristic approaches, evaluation datasets, and qualitative examples to illustrate its effectiveness in reducing token count while maintaining semantic integrity. Various aspects related to token optimization, including adjusting spaces and capitalizations, replacing synonyms, lemmatization, bracket removal, handling compound words, stop word removal, punctuation removal, and handling acronyms, are discussed. These heuristic approaches are implemented using tools and resources such as NLTK for lemmatization, thesaurus dictionaries for synonym replacement, and spell checkers for enhancing accuracy. Additionally, the evaluation of token optimization methods involves assessing their impact on different tasks through evaluation metrics like accuracy and compression percentage. Diverse datasets, such as CosmosQA, LogiQA, ReCLoR, ConTRoL, Dataset I, and Dataset II, cover various tasks such as question answering and summarization, providing contexts for evaluating token optimization techniques. Qualitative examples demonstrate how sentence simplification can condense sentences while preserving their core meaning, showcasing the readability and efficiency enhancements achieved through token optimization.[22]

Saurabh Deochake et al Introduced importance of adopting cost-effective strategies in cloud computing, such as leveraging different pricing models, optimizing resource allocation, and implementing storage optimization techniques, is highlighted in the Cloud Cost Optimization paper by Saurabh Deochake. Cloud computing has transformed businesses with benefits like scalability and cost-effectiveness, but managing costs is essential, with factors such as resource allocation and pricing models playing a key role. Efficient strategies like right-sizing, autoscaling, using spot instances, and leveraging reserved instances can lead to substantial cost savings by aligning resources with actual needs and utilizing discounts effectively. Additionally, serverless computing and containerization offer flexible and efficient solutions to optimize costs based on actual resource usage, providing granular control and scalable options. In terms of storage optimization, data deduplication, compression, and data lifecycle management policies are vital techniques to reduce storage costs.Implementing data lifecycle policies automates data migration to cost-effective storage tiers based on access frequency, optimizing storage expenses without compromising data accessibility or integrity. These strategies ensure that data is stored in suitable storage options according to its value and access patterns, aligning storage costs with data usage and value. Overall, significant cost savings, enhanced operational efficiency, and maintaining competitiveness in the modern cloud computing landscape can be achieved by understanding and effectively implementing these strategies.[23]

Security any application software including recommendation system is equally important as designing a system [24].

3. PROBLEM STATEMENT

The primary problem associated with cloud computing for new users is twofold: cost management and the challenge of selecting appropriate services. New users often find it challenging to manage and control the costs associated with cloud services and might end up with high bills in the confusing "pay-as-you-go" model if proper consideration is not given. While considering a small e-commerce startup. It is required to host their website and databases on a cloud platform. During a holiday sale, the website experiences a sudden surge in traffic. Without proper resource monitoring and auto-scaling it may required to inadvertently provision excessive resources, resulting in a significant increase in costs that they didn't anticipate. Secondly, choosing the right combination of cloud services is another significant challenge. New users may struggle to select services that best fit their specific needs. Imagine a data analytics company that wants to perform complex data processing tasks. Choosing the incorrect service can result in inefficiency and increased expenses. To tackle these challenges successfully, it's essential for newcomers to develop a thorough grasp of cost management techniques and utilize tools and knowledge to pick the most appropriate services for their individual requirements.

4. PROPOSED SYSTEM

New users stepping into cloud computing often face two significant challenges**:** effectively **managing costs and navigating the array of service options available.** The complexities of the "pay-as-you-go" model can confuse users, leading to unforeseen expenses. Imagine a scenario where a growing e-commerce business sees a surge in website traffic during a seasonal sale; without careful resource monitoring, they risk provisioning too much and facing unexpected costs. Consider a local bookstore looking to expand its reach by launching an online store. As they start receiving more orders than expected, they're unsure about which cloud services they need to manage the increased website traffic effectively. Additionally, they're concerned about the potential costs associated with these services, given their limited knowledge and experience in the realm of cloud computing.

To tackle these challenges, a holistic solution is proposed. This solution involves creating user-friendly interfaces with comprehensive forms to gather necessary information. By leveraging sophisticated algorithms, such as the **Large Language Model (LLM), trained on relevant cloud service data**, the system can provide well-informed recommendations. By organizing prompts and integrating user input collected from the forms, the solution makes it easier to **generate accurate cost estimates and personalized service suggestions.**

Additionally, the **solution provides detailed steps to seamlessly connect all recommended AWS services**. These step-by-step instructions ensure a smooth implementation process, guiding users through the configuration and integration of each service. By offering clear guidance on how to set up and connect the recommended services, users can effectively **leverage the suggested cloud infrastructure to meet their specific needs**. This comprehensive approach aims to empower users with the knowledge and resources needed to make informed decisions and successfully deploy their cloud solutions on the AWS platform.

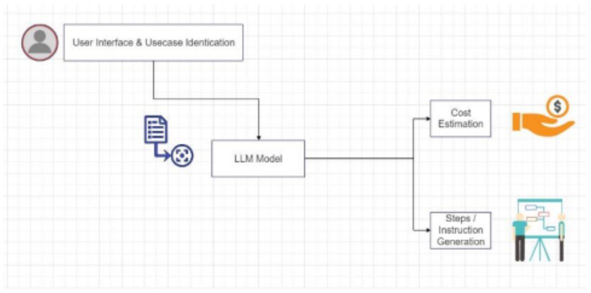

User Interface & Usecase Identication
LLM Model
Cost Estimation
Steps / Instruction Generation

Fig.1. Architecture design

The fig.1 depicts a recommendation system for cloud services which involves four modules such as,

User Interface & Usecase Identification: The initial step involves collecting information on user input and identifies the specific use case for which a cloud service recommendation is needed.

Large Language Model: Large Language Model(LLM) , which is a type of artificial intelligence (AI) that's trained on a massive amount of data. In the context of this diagram, the LLM model likely analyzes the user input and use case to generate potential cloud service recommendations.

Cost Estimation: LLM assists in selecting relevant AWS cloud services based on refined requirements, considering factors such as scalability, data storage needs, and security requirements. LLM helps estimate development and operational costs on AWS, including the price of each AWS service, required computing resources (e.g., virtual machines), and data storage requirements.

Generating Steps to Connect Services: LLM assists in outlining technical steps to integrate chosen AWS services and provides detailed guidance on connecting services for desired functionality.

4.1.METHODOLOGY

The methodology primarily centers around prompt engineering, a critical process in effectively utilizing a Large Language Model (LLM). Through prompt engineering, prompts are refined to guide the LLM in various tasks. Initially, requirements are gathered by crafting prompts that extract detailed information from stakeholders and users. These prompts are used by the LLM to analyze data, identify missing elements, and ensure clarity. Furthermore, prompts direct the LLM in selecting relevant Amazon Web Services (AWS), estimating costs, and generating integration steps. By emphasizing prompt engineering, the capabilities of the LLM are optimized, enabling it to serve as a valuable assistant throughout the process. This approach underscores a commitment to leveraging advanced AI techniques while integrating human expertise in decision-making and design.

4.2. SECTION DESCRIPTION

Undertaking is typically divided into four main modules based on the way it represents the outcome. These modules are:

- User interface Development
- Leveraging LLM for Cloud Service Recommendations
- Cost estimation module
- AWS Services Integration

4.2.1. *USER INTERFACE DEVELOPMENT*

A user interface is established using React JS to gather requirements. React JS is a component-based architecture which has ability to efficiently develop the user interface as users interact with the system are essential for creating a smooth and engaging experience. It begins with the initiation of the registration and login processes. Subsequently, the user is presented with a homepage where the selection of the domain for their use case is facilitated. Based on the chosen domain, a set of questions is provided to be answered by the user. These questioned are framed to understand the use-case of the user. In this manner, the user's use case is collected and compiled in the form of JSON/XML. This compiled data is then employed as input for the generation of the required services and the estimation of associated costs.

4.2.2.LEVERAGING LLM FOR CLOUD SERVICE RECOMMENDATIONS

In the utilization of language models like OpenAI's Generative Pre-trained Transformers 3 and 4, Palm2, and Gemini-pro, the structuring of the prompt is deemed essential for the attainment of meaningful and contextually relevant responses. The initiation of the language model underlying the agent by the user's provided prompt aids in the generation of appropriate responses to inquiries. The utilization of an external Large Language Model (LLM) facilitates the generation of service recommendations. LLMs such as Generative Pre-trained Transformer-3 or Generative Pre-trained Transformer-4, Palm2, and Gemini-pro are employed for leveraging large text data to create human-like text for tasks such as generating text, language translation, and summarising information.

Structure of the Prompt:

The prompt is broadly structured into two sections:
Role: The expected behavior of the agent and an explanation of the concept of tools.
Instructions: Provision of examples of tasks and their solutions.
The integration of an external LLM facilitates the generation of service recommendations and cost estimates, empowering users to make well-informed decisions in their cloud adoption journey.

a) Different Kinds of LLM

Language Learning Models (LLMs) represent advanced software engineered to grasp and interpret human language and intricate datasets. These systems undergo rigorous training using extensive data, often sourced from the Internet, spanning thousands to millions of gigabytes of textual information. However, the effectiveness of LLMs hinges significantly on the quality of the data samples utilized. To optimize learning, developers may opt for meticulously curated datasets, ensuring enhanced comprehension of natural language nuances and complexities.

Various iterations of large language models (LLMs) have been developed, each serving distinct purposes within the field. These include Gemini, Bard, Palm2, GPT 3.5, GPT 4, and Mistral 8x7b.

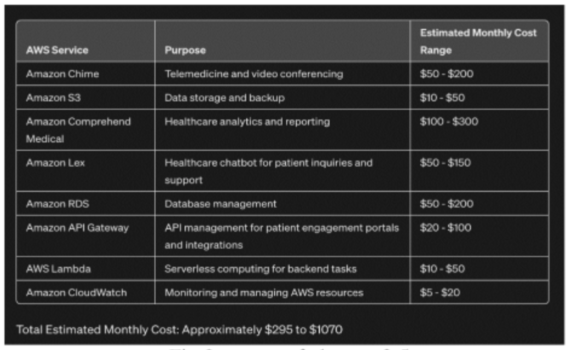
b) Exploring the Responses: Testing Different LLMs

ChatGPT- 3.5

GPT-3.5 stands as a refined iteration of OpenAI's GPT-3 model, showcasing enhancements and refinements aimed at boosting its effectiveness, speed, and abilities.

Overall, GPT-3.5 represents a step forward in natural language processing technology, offering improved performance, capabilities, and reliability compared to previous iterations. These advancements contribute to its broader applicability across various domains and applications, ranging from conversational agents and content generation to language translation and sentiment analysis.

The output of cloud services recommendations for Healthcare is shown in the given figure (Fig- 2), generated by ChatGPT-3.5



AWS Service	Purpose	Estimated Monthly Cost Range
Amazon Chime	Telemedicine and video conferencing	$50 - $200
Amazon S3	Data storage and backup	$10 - $50
Amazon Comprehend Medical	Healthcare analytics and reporting	$100 - $300
Amazon Lex	Healthcare chatbot for patient inquiries and support	$50 - $150
Amazon RDS	Database management	$50 - $200
Amazon API Gateway	API management for patient engagement portals and integrations	$20 - $100
AWS Lambda	Serverless computing for backend tasks	$10 - $50
Amazon CloudWatch	Monitoring and managing AWS resources	$5 - $20

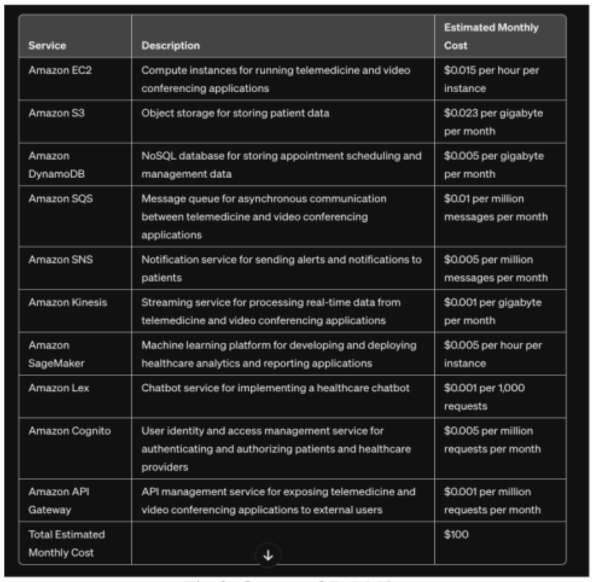
Total Estimated Monthly Cost: Approximately $295 to $1070

Fig.2. output of chatgpt-3.5

Fig.2 depicts cloud services recommendations for Healthcare generated by ChatGPT-3.5. Without understanding the healthcare context, GPT-3.5 might focus on general AWS services such ad S3, Cloudwatch and lambda and their estimated monthly costs. This wouldn't be helpful for pinpointing healthcare-specific services and their pricing structures. GPT-3.5 might not have the specific knowledge about healthcare services like Amazon HealthLake or Transcribe Medical. This means it wouldn't be able to explain their functionalities or how they benefit healthcare applications. By not considering the healthcare context and offering generic information, GPT-3.5's response wouldn't be as useful as one that targets healthcare needs and provides specific details on relevant services and their pricing.

PaLM2

Pathways Language Model 2 (PaLM 2), renowned for its advancements, was among Google's most sophisticated large language models. Its extensive training on a vast dataset spanning over 100 languages endowed it with a nuanced understanding of diverse cultural nuances, enabling proficient translation and generation of text, including intricate literary forms such as poems, riddles, and idioms. Notably, PaLM 2 exhibited enhanced logical reasoning capabilities, facilitating a deeper comprehension of inter-idea relationships and adept problem-solving. Additionally, its unique feature allowed for code generation and debugging, rendering it invaluable for programmers. However, PaLM 2 has since been superseded by Gemini, Google's current flagship LLM, which offers comparable functionalities alongside further advancements.

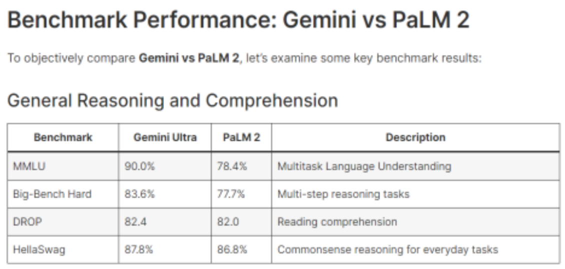


Service	Description	Estimated Monthly Cost
Amazon EC2	Compute instances for running telemedicine and video conferencing applications	$0.015 per hour per instance
Amazon S3	Object storage for storing patient data	$0.023 per gigabyte per month
Amazon DynamoDB	NoSQL database for storing appointment scheduling and management data	$0.005 per gigabyte per month
Amazon SQS	Message queue for asynchronous communication between telemedicine and video conferencing applications	$0.01 per million messages per month
Amazon SNS	Notification service for sending alerts and notifications to patients	$0.005 per million messages per month
Amazon Kinesis	Streaming service for processing real-time data from telemedicine and video conferencing applications	$0.001 per gigabyte per month
Amazon SageMaker	Machine learning platform for developing and deploying healthcare analytics and reporting applications	$0.005 per hour per instance
Amazon Lex	Chatbot service for implementing a healthcare chatbot	$0.001 per 1,000 requests
Amazon Cognito	User identity and access management service for authenticating and authorizing patients and healthcare providers	$0.005 per million requests per month
Amazon API Gateway	API management service for exposing telemedicine and video conferencing applications to external users	$0.001 per million requests per month
Total Estimated Monthly Cost		$100

Fig.3. Output of PaLM2

The output of cloud services recommendations for Healthcare is shown in the given figure (Fig- 3), generated by PaLM2.

Like GPT-3.5, PaLM 2 might prioritize explaining the general functionalities of AWS services in the table (EC2, S3, DynamoDB) instead of their specific applications in healthcare. When explaining services like Amazon Kinesis or SageMaker, PaLM 2's response might lack the precision needed to convey their use cases in processing real-time healthcare data or developing healthcare analytics applications.



Benchmark Performance: Gemini vs PaLM 2

To objectively compare **Gemini vs PaLM 2**, let's examine some key benchmark results:

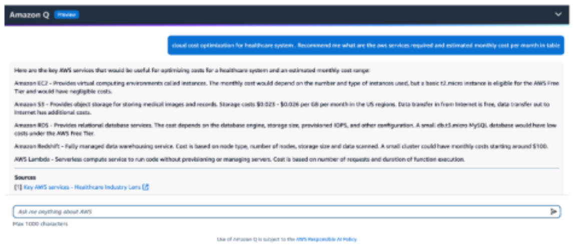
General Reasoning and Comprehension

Benchmark	Gemini Ultra	PaLM 2	Description
MMLU	90.0%	78.4%	Multitask Language Understanding
Big-Bench Hard	83.6%	77.7%	Multi-step reasoning tasks
DROP	82.4	82.0	Reading comprehension
HellaSwag	87.8%	86.8%	Commonsense reasoning for everyday tasks

Fig.4. Compare Gemini vs PaLM

Amazon Q

Deep expertise in AWS services, best practices, and documentation is incorporated into Amazon Q, rendering it a valuable resource for businesses aiming to boost employee productivity and streamline tasks within the AWS cloud environment. With seamless integration with company data sources and internal systems, tailored conversations and problem-solving capabilities are facilitated, ensuring relevance to specific business needs. Moreover, security and privacy considerations have been prioritized in its development, aligning with the sensitivity of business information. Notably, users can leverage Amazon Q to seek recommendations on AWS services, further enhancing its utility as a comprehensive AI assistant tailored to AWS users' requirements.


Amazon Q
Preview
Ask me anything about AWS
Max 1000 characters
Use of Amazon Q is subject to the AWS Responsible AI Policy

cloud cost optimization for healthcare system . Recommend me what are the aws services required and estimated monthly cost per month in table

Here are the key AWS services that would be useful for optimizing costs for a healthcare system and an estimated monthly cost range:

Amazon EC2 - Provides virtual computing environments called instances. The monthly cost would depend on the number and type of instances used, but a basic t2.micro instance is eligible for the AWS Free Tier and would have negligible costs.

Amazon S3 - Provides object storage for storing medical images and records. Storage costs $0.023 - $0.026 per GB per month in the US regions. Data transfer in from Internet is free, data transfer out to Internet has additional costs.

Amazon RDS - Provides relational database services. The cost depends on the database engine, storage size, provisioned IOPS, and other configuration. A small db.t3.micro MySQL database would have low costs under the AWS Free Tier.

Amazon Redshift - Fully managed data warehousing service. Cost is based on node type, number of nodes, storage size and data scanned. A small cluster could have monthly costs starting around $100.

AWS Lambda - Serverless compute service to run code without provisioning or managing servers. Cost is based on number of requests and duration of function execution.

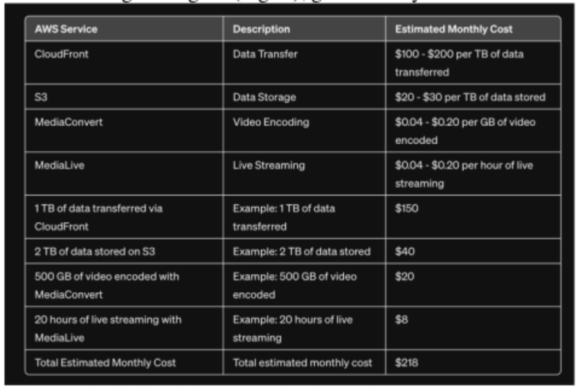
Sources
[1] Key AWS services - Healthcare Industry Lens

Fig.5. Sevice and cost Output of Amazon Q

The drawback of Amazon Q is that it is in preview and is limited to a user interface only, lacking an API.

GEMINI

Google AI's Gemini Large Language Model stands as a robust language model, honed through extensive training on a wide-ranging corpus of text and code. This training equips it with the capacity to comprehend and produce human-like text, translate between languages, and manage various data formats such as text, code, images, and video. Notably, Gemini LLM demonstrates exceptional reasoning capabilities and can scale effectively, offering three distinct versions: Ultra, Pro, and Nano. While it may lag behind GPT-4 in certain aspects, it maintains an advantage in grasping common sense.
The output of cloud services recommendations for Healthcare is shown in the given figure (Fig- 6), generated by Gemini 1.5 Pro



AWS Service	Description	Estimated Monthly Cost
CloudFront	Data Transfer	$100 - $200 per TB of data transferred
S3	Data Storage	$20 - $30 per TB of data stored
MediaConvert	Video Encoding	$0.04 - $0.20 per GB of video encoded
MediaLive	Live Streaming	$0.04 - $0.20 per hour of live streaming
1 TB of data transferred via CloudFront	Example: 1 TB of data transferred	$150
2 TB of data stored on S3	Example: 2 TB of data stored	$40
500 GB of video encoded with MediaConvert	Example: 500 GB of video encoded	$20
20 hours of live streaming with MediaLive	Example: 20 hours of live streaming	$8
Total Estimated Monthly Cost	Total estimated monthly cost	$218

Fig.6. Gemini output of estimated cost

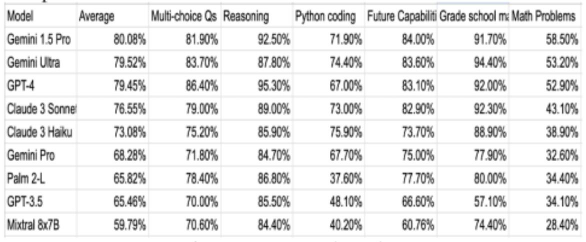
From fig.6 Understands the context of healthcare and prioritizes explaining services relevant to the field (HealthLake,Comprehend Medical, Transcribe Medical).

Possesses a deeper knowledge base of healthcare data storage, analysis, and management tools offered by AWS. Delivers a response directly addressing healthcare needs. It explains relevant services, their functionalities in a healthcare context, and provides concrete information on pricing, making it easier to evaluate their usefulness and cost-effectiveness. In conclusion, Gemini's focus on healthcare, deeper knowledge of healthcare-related AWS services, and provision of actionable information on pricing and resource make it a superior choice for understanding these services compared to GPT-3.5 and PaLM 2.

Thus, based on the outputs generated in Fig 4.2, 4.3 and 4.6 by ChatGPT 3.5, PaLM2 and Gemini 1.5 Pro respectively, it has been concluded that Gemini 1.5 Pro is more suitable for the usecase.

c) LLM Benchmarking:

Fig. 7 showcases benchmark results, providing a comprehensive visual representation of LLM's performance. This demonstrates that the performance of Gemini 1.5 Pro is superior compared to all other LLMs.



Model	Average	Multi-choice Qs	Reasoning	Python coding	Future Capabiliti	Grade school m	Math Problems
Gemini 1.5 Pro	80.08%	81.90%	92.50%	71.90%	84.00%	91.70%	58.50%
Gemini Ultra	79.52%	83.70%	87.80%	74.40%	83.60%	94.40%	53.20%
GPT-4	79.45%	86.40%	95.30%	67.00%	83.10%	92.00%	52.90%
Claude 3 Sonne	76.55%	79.00%	89.00%	73.00%	82.90%	92.30%	43.10%
Claude 3 Haiku	73.08%	75.20%	85.90%	75.90%	73.70%	88.90%	38.90%
Gemini Pro	68.28%	71.80%	84.70%	67.70%	75.00%	77.90%	32.60%
Palm 2-L	65.82%	78.40%	86.80%	37.60%	77.70%	80.00%	34.40%
GPT-3.5	65.46%	70.00%	85.50%	48.10%	66.60%	57.10%	34.10%
Mixtral 8x7B	59.79%	70.60%	84.40%	40.20%	60.76%	74.40%	28.40%

Fig.7. LLM Benchmarks

From the fig 7 it is observed that Gemini 1.5 Pro's average score is the highest across all measured tasks. Particularly, it excels at multiple choice questions and resolving math problems. This suggests that Gemini 1.5 Pro has a strong ability to analyze information and select the best answer among options, along with a deep understanding of mathematical concepts and problem-solving techniques. Therefore Gemini 1.5 pro has been considered as appropriate solution.

2) *COST ESTIMATION*

User information, obtained in XML or JSON format, is incorporated into the prompt. The structuring of the prompt encompasses several essential components, including: The formation of use case requirements is initiated in the initial part of the prompt, where the user's specific needs, project details, and preferences are outlined based on the provided XML or JSON data. Subsequently, the prompt specifies the services recommended to meet the user's requirements, elucidating the monthly expenses associated with each suggested service. The concept of optimized cost is addressed in the prompt, with an explanation of how service recommendations prioritize cost savings and operational efficiency. The prompt proceeds to provide the total estimated cost, aggregating the monthly costs of all recommended services. Furthermore, the prompt emphasizes the benefits of the recommended services, highlighting improvements in performance, scalability, and cost-effectiveness. Upon processing the input data and generating relevant recommendations and cost estimates, the output is formatted into a PDF document. The user is then furnished with this PDF document, which includes a comprehensive report detailing their use case requirements, service recommendations, cost breakdown, information on optimized costs, total cost, and benefits. This format ensures that the user receives a structured and easily accessible report based on their input data.

3) *AWS SERVICES INTEGRATION*

The detailed step-by-step guide for seamlessly connecting and utilizing various AWS services is provided by the AWS Services Integration module. Each step is meticulously outlined, ensuring a smooth integration process and enabling users to leverage the capabilities of AWS services effectively within their projects. The steps are generated through an API call by the Gemini Pro LLM (Large Language Model), relying on the input provided to it. The input is processed, and the context is analyzed by the Gemini Pro LLM. Based on its training on a vast dataset, which includes information about AWS services and their integration, coherent and relevant step-by-step instructions for connecting these services are generated. These generated steps, along with any additional details or links, are then returned as a response from the Gemini Pro LLM API call. The generated steps can be reviewed by users to ensure they meet their requirements.

B. *RESULT AND DISCUSSION*

In the methodology section, the focus is shifted to the design and development phases. The utilization of React's component-based architecture and the customization capabilities of React forms are highlighted, demonstrating the strategic approach taken in this article.

Distinct endpoints for registration, login, and a home page have been developed in our application. The home endpoint allows users to select forms specific to their domain of interest. The domains covered include healthcare, education, e-commerce, media-entertainment, and financial services. This structure ensures a tailored experience for users across various sectors.

Fig.8 shows the creation of user accounts involves a straightforward process where users are asked to furnish imply details. Upon accessing the application's registration page, users are guided through a series of prompts that request their name, email address, and password. These pieces of information are crucial for establishing a unique identity for each user within the system.

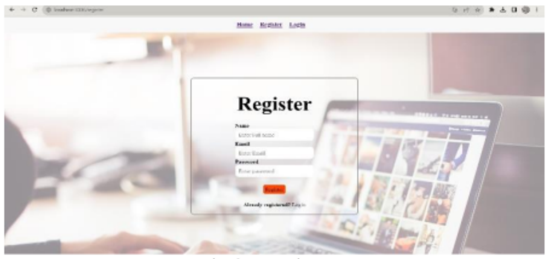


Fig.8. Register page

Fig.9 Describes the User credentials: Registered users are required to input their unique login credentials, typically consisting of a username, email address, along with their associated password.

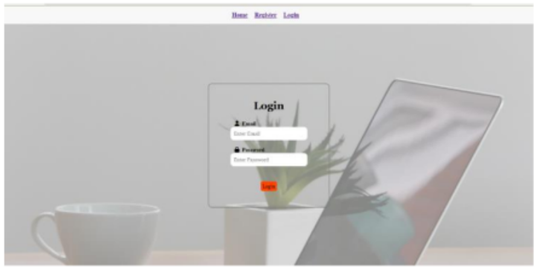


Fig.9. Login page

Fig.10 and Fig.11, It is a starting point for navigation and interaction. Here, the user is provided with few domains such as Healthcare, Ecommerce, Education, Financial Services, Media and Entertainment. From this, the user need to choose their domain. After selecting domain, user will get redirected to a form where they are required to fill a questionnaire to get their recommendation.

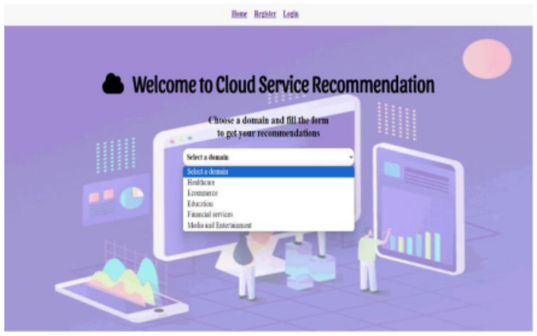
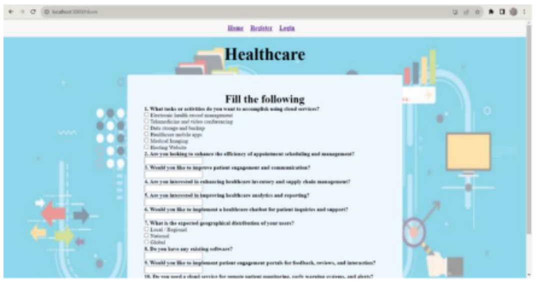


Fig.10. Home page


Home Register Login

Healthcare

Fill the following

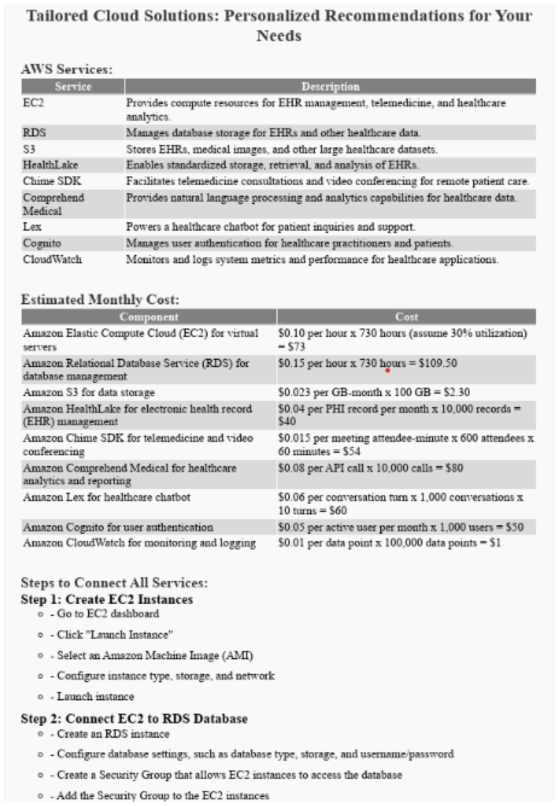
1. What tasks or activities do you want to accomplish using cloud services?
- ☐ Electronic health record management
- ☐ Telemedicine and video conferencing
- ☐ Data storage and backup
- ☐ Healthcare mobile apps
- ☐ Medical Imaging
- ☐ Hosting Website

2. Are you looking to enhance the efficiency of appointment scheduling and management?
3. Would you like to improve patient engagement and communication?
4. Are you interested in enhancing healthcare inventory and supply chain management?
5. Are you interested in improving healthcare analytics and reporting?
6. Would you like to implement a healthcare chatbot for patient inquiries and support?
7. What is the expected geographical distribution of your users?
- ☐ Local / Regional
- ☐ National
- ☐ Global

8. Do you have any existing software?
9. Would you like to implement patient engagement portals for feedback, reviews, and interaction?
10. Do you need a cloud service for remote patient monitoring, early warning systems, and alerts?

Fig.11. Healthcare form

Fig 12 and 13 show the output of cloud services and the estimated cost of the services and the total cost per month for the user usecase. And Steps to connect the Cloud services.


Tailored Cloud Solutions: Personalized Recommendations for Your Needs

AWS Services:

Service	Description
EC2	Provides compute resources for EHR management, telemedicine, and healthcare analytics.
RDS	Manages database storage for EHRs and other healthcare data.
S3	Stores EHRs, medical images, and other large healthcare datasets.
HealthLake	Enables standardized storage, retrieval, and analysis of EHRs.
Chime SDK	Facilitates telemedicine consultations and video conferencing for remote patient care.
Comprehend Medical	Provides natural language processing and analytics capabilities for healthcare data.
Lex	Powers a healthcare chatbot for patient inquiries and support.
Cognito	Manages user authentication for healthcare practitioners and patients.
CloudWatch	Monitors and logs system metrics and performance for healthcare applications.

Estimated Monthly Cost:

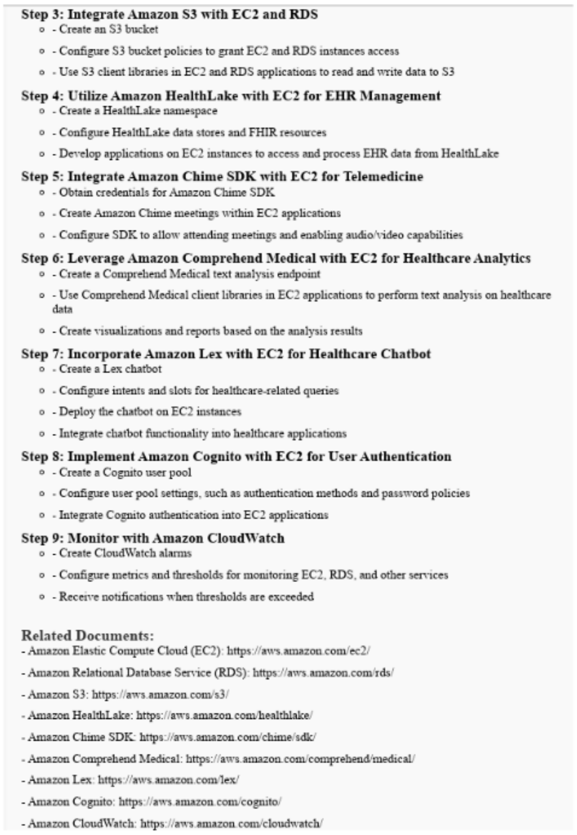
Component	Cost
Amazon Elastic Compute Cloud (EC2) for virtual servers	\$0.10 per hour x 730 hours (assume 30% utilization) = \$73
Amazon Relational Database Service (RDS) for database management	\$0.15 per hour x 730 hours = \$109.50
Amazon S3 for data storage	\$0.023 per GB-month x 100 GB = \$2.30
Amazon HealthLake for electronic health record (EHR) management	\$0.04 per PHI record per month x 10,000 records = \$40
Amazon Chime SDK for telemedicine and video conferencing	\$0.015 per meeting attendee-minute x 600 attendees x 60 minutes = \$54
Amazon Comprehend Medical for healthcare analytics and reporting	\$0.08 per API call x 10,000 calls = \$80
Amazon Lex for healthcare chatbot	\$0.06 per conversation turn x 1,000 conversations x 10 turns = \$60
Amazon Cognito for user authentication	\$0.05 per active user per month x 1,000 users = \$50
Amazon CloudWatch for monitoring and logging	\$0.01 per data point x 100,000 data points = \$1

Steps to Connect All Services:

Step 1: Create EC2 Instances
- - Go to EC2 dashboard
- - Click "Launch Instance"
- - Select an Amazon Machine Image (AMI)
- - Configure instance type, storage, and network
- - Launch instance

Step 2: Connect EC2 to RDS Database
- - Create an RDS instance
- - Configure database settings, such as database type, storage, and username/password
- - Create a Security Group that allows EC2 instances to access the database
- - Add the Security Group to the EC2 instances

Fig.12. Output of Cloud services and Cost of the services needs for the usecase.


Step 3: Integrate Amazon S3 with EC2 and RDS
- - Create an S3 bucket
- - Configure S3 bucket policies to grant EC2 and RDS instances access
- - Use S3 client libraries in EC2 and RDS applications to read and write data to S3

Step 4: Utilize Amazon HealthLake with EC2 for EHR Management
- - Create a HealthLake namespace
- - Configure HealthLake data stores and FHIR resources
- - Develop applications on EC2 instances to access and process EHR data from HealthLake

Step 5: Integrate Amazon Chime SDK with EC2 for Telemedicine
- - Obtain credentials for Amazon Chime SDK
- - Create Amazon Chime meetings within EC2 applications
- - Configure SDK to allow attending meetings and enabling audio/video capabilities

Step 6: Leverage Amazon Comprehend Medical with EC2 for Healthcare Analytics
- - Create a Comprehend Medical text analysis endpoint
- - Use Comprehend Medical client libraries in EC2 applications to perform text analysis on healthcare data
- - Create visualizations and reports based on the analysis results

Step 7: Incorporate Amazon Lex with EC2 for Healthcare Chatbot
- - Create a Lex chatbot
- - Configure intents and slots for healthcare-related queries
- - Deploy the chatbot on EC2 instances
- - Integrate chatbot functionality into healthcare applications

Step 8: Implement Amazon Cognito with EC2 for User Authentication
- - Create a Cognito user pool
- - Configure user pool settings, such as authentication methods and password policies
- - Integrate Cognito authentication into EC2 applications

Step 9: Monitor with Amazon CloudWatch
- - Create CloudWatch alarms
- - Configure metrics and thresholds for monitoring EC2, RDS, and other services
- - Receive notifications when thresholds are exceeded

Related Documents:
- Amazon Elastic Compute Cloud (EC2): https://aws.amazon.com/ec2/
- Amazon Relational Database Service (RDS): https://aws.amazon.com/rds/
- Amazon S3: https://aws.amazon.com/s3/
- Amazon HealthLake: https://aws.amazon.com/healthlake/
- Amazon Chime SDK: https://aws.amazon.com/chime/sdk/
- Amazon Comprehend Medical: https://aws.amazon.com/comprehend/medical/
- Amazon Lex: https://aws.amazon.com/lex/
- Amazon Cognito: https://aws.amazon.com/cognito/
- Amazon CloudWatch: https://aws.amazon.com/cloudwatch/

Fig.13. Output for Steps to connect the Cloud services

5. CONCLUSION

In conclusion, it is recognized that a recommendation system for new users is considered a fundamental component of the Article, designed with the primary objective of streamlining the process of selecting and adopting cloud services. Within this framework, a user-friendly interface is facilitated, allowing for the creation of user profiles and the collection of vital input information. At its core lies a sophisticated recommendation engine, which is utilized to deliver tailored and optimized solutions to users. However, it is acknowledged that there are two main drawbacks that need addressing. Firstly, high response times are experienced by users, averaging 15 seconds or more, which can lead to frustration. One potential solution to this issue involves the implementation of caching techniques to mitigate latency problems and enhance system responsiveness. Secondly, while an accuracy rate of 80-85% is achieved by our recommendation engine, there remains room for improvement. Incorporating web scraping functionality into the system to gather information from authoritative sources such as official cloud optimizing documents, AWS whitepapers, and system design documents can enrich the recommendation process, leading to more precise and relevant suggestions for users, thereby improving overall accuracy. Despite these challenges, the system aims to empower new users, rendering cloud services more accessible and user-friendly, ultimately equipping users with the knowledge required to make informed choices and maximize the effective utilization of cloud resources.

REFERENCES

[1] Kalaiselvi, R, Kousalya, K, "Statistical modelling and parametric optimization in document fragmentation:, Neural Computing and Applications, pp.1-10, 2019.

[2] Dehua Kong, Yuqing Zhai, "Trust Based Recommendation System in Service-oriented Cloud Computing", DOI: 10.1109/CSC.2012.34, Published: 2012 International Conference on Cloud and Service Computing, Publisher: IEEE

[3] Chao Luo, Xiaoqiang Liu, Kai Zhang, Qinghong Chang, "A Recommendation System for Cloud Services Based on Knowledge Graph", DOI: 10.1109/ICSESS.2018.8663716, Published: 2018 IEEE 9th International Conference on Software Engineering and Service Science (ICSESS), Publisher: IEEE

[4] Rui-dong Qi, Jian-tao Zhou, Zhuowei Wang, Xiaoyu Song, "An elastic recommender process for cloud service recommendation scalability", DOI: 10.1002/cpe.7066, Published: 28 May 2022, Publisher: Wiley

[5] Abid Mahmood, Umar Shoaib, M. Shahzad Sarfraz, "A Recommendation System for Cloud Services Selection based on Intelligent Agents", DOI: 10.17485/ijst/ 2018/ v11i9/ 119843, Published: March 2018 Indian Journal of Science and Technology

[6] Rahma DJIROUN, Meriem Amel GUESSOUM, Kamel BOUKHALFA, Elhadj BENKHELIFA, "A Novel Cloud Services Recommendation System Based on Automatic Learning Techniques, DOI: 10.1109/ICTCS.2017.58, Published: 2017 International Conference on New Trends in Computing Sciences (ICTCS), Publisher: IEEE

[7] Wenqi Fan, Zihuai Zhao, Jiatong Li, Yunqing Liu, Xiaowei Mei, Yiqi Wang, Zhen Wen, Fei Wang, Xiangyu Zhao, Jiliang Tang, Qing Li, "Recommender Systems in the Era of Large Language Models (LLMs)", DOI: 10.48550/arXiv.2307.02046, Published: Wed, 5 Jul 2023

[8] Likang Wu, Zhi Zheng, Zhaopeng Qiu, Hao Wang, Hongchao Gu, Tingjia Shen, Chuan Qin, Chen Zhu, Hengshu Zhu, Qi Liu, Hui Xiong, Enhong Chen, "A Survey on Large Language Models for Recommendation", DOI: 10.48550/arXiv.2305.19860, Published: Wed, 31 May 2023, Publisher: arXiv

[9] An Zhang, Leheng Sheng, Yuxin Chen, Hao Li, Yang Deng, Xiang Wang, Tat-Seng Chua, "On Generative Agents in Recommendation", DOI: 10.48550/arXiv.2310.10108, Published: Mon, 16 Oct 2023, Publisher: arXiv

[10] Nidhi Hegde, Sujoy Paul, Gagan Madan, Gaurav Aggarwal, "Analyzing the Efficacy of an LLM-Only Approach for Image-based Document Question Answering", DOI: 10.48550/arXiv.2309.14389, Published: Mon, 25 Sep 2023, Publisher: arXiv

[11] Liangmin Guo, Kaixuan Luan, Xiaoyao Zheng, Jing Qian, "A Service Recommendation Method Based on Requirements for the Cloud Environment", DOI: 10.1155/2021/6669798, Published: 25 Mar 2021, Publisher: Hindawi

[12] Tao Fan, Yan Kang, Guoqiang Ma, Weijing Chen, Wenbin Wei, Lixin Fan, Qiang Yang "FATE-LLM: A Industrial Grade Federated Learning Framework for Large Language Models", DOI: 10.48550/arXiv.2310.10049, Published : 16 Oct 2023,Publisher : arXiv

[13] Wenxuan Zhang, Hongzhi Liu, Yingpeng Du, Chen Zhu, Yang Song, Hengshu Zhu, Zhonghai Wu "Bridging the Information Gap Between Domain-Specific Model and General LLM for Personalized Recommendation"DOI:10.48550/arXiv.2311.03778,Published:7 Nov2023,Publisher:arXiv

[14] Junjie Zhang, Ruobing Xie, Yupeng Hou, Wayne Xin Zhao, Leyu Lin, Ji-Rong Wen "Recommendation as Instruction Following: A Large Language Model Empowered RecommendationApproach"DOI:10.48550/arXiv.2305.07001,Published:11 May 2023,Publisher:arXiv

[15] Miranda Zhang, Rajiv Ranjan, Michael Menzel, Surya Nepal, Peter Strazdins, Lizhe Wang," A Cloud Infrastructure Service Recommendation System for Optimizing Real-time QoS Provisioning Constraints" DOI:10.48550/arXiv.1504.01828,Published: 8 Apr 2015 , Publisher: arXiv

[16] Lanning Wei, Zhiqiang He, Huan Zhao, Quanming Yao," Unleashing the Power of Graph Learning through LLM-based Autonomous Agents" DOI: 10.48550/arXiv.2309.04565,Published : 8 Sep 2023,Publisher: arXiv

[17] Daixing Zhong , Gangjun Yang , Jiashuang Fan, Baozhen Tian, Yukun Zhang, DOI: 10.1016/j.csi.2022.103632, Published: 10 September 2020, Publisher: ScienceDirect

[18] Xu Huang, Jianxun Lian, Yuxuan Lei, Jing Yao, Defu Lian, Xing Xie, "Recommender AI Agent: Integrating Large Language Models for Interactive Recommendations", DOI: 10.48550/arXiv.2308.16505, Published: 31 Aug 2023, Publisher: arXiv

[19] Yunfan Gao, Tao Sheng, Youlin Xiang, Yun Xiong, Haofen Wang, Jiawei Zhang, "Chat-REC: Towards Interactive and Explainable LLMs-Augmented Recommender System", DOI: 10.48550/arXiv.2303.14524, Published: 25 Mar 2023, Publisher: arXiv

[20] Keqin Bao, Jizhi Zhang, Yang Zhang, Wenjie Wang, Fuli Feng, Xiangnan He, "TALLRec: An Effective and Efficient Tuning Framework to Align Large Language Model with Recommendation", DOI: 10.1145/3604915.3608857, Published: 30 Apr 2023, Publisher: arXiv

[21] Junling Liu, Chao Liu, Peilin Zhou, Qichen Ye, Dading Chong, Kang Zhou, Yueqi Xie, Yuwei Cao, Shoujin Wang, Chenyu You, Philip S.Yu, "LLMRec: Benchmarking Large Language Models on Recommendation Task", DOI: 10.48550/arXiv.2308.12241, Published: 23 Aug 2023, Publisher: arXiv.

[22] Shivanshu Shekhar, Tanishq Dubey, Koyel Mukherjee, Apoorv Saxena, Atharv Tyagi, Nishanth Kotla ," Towards Optimizing the Costs of LLM Usage", DOI: 10.48550/arXiv.2402.01742, Published: 29 Jan 2024, Publisher: arXiv.

[23] Saurabh Deochake "Cloud Cost Optimization: A Comprehensive Review of Strategies and Case Studies", DOI: 10.48550/arXiv.2307.12479, 24 Jul 2023, Publisher: arXiv.

[24] Uloli, Thiruvaazhi, G. Sudha Sadasivam, and Arthi R. "Requirements Analysis of Security and Privacy of Mobile Payments Indian Context." *International Journal of Mobile Communications* 20 (6): 639–58, 2022.

Plag cloud

ORIGINALITY REPORT

3%	2%	3%	1%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	arxiv.org Internet Source	1%
2	Submitted to University of Malaya Student Paper	<1%
3	Thiruvaazhi Uloli, Sowmiya K, Vedhakalaivani P, Nathiyaa R. "A Survey on Bitcoin Deanonymization", 2023 2nd International Conference on Advancements in Electrical, Electronics, Communication, Computing and Automation (ICAECA), 2023 Publication	<1%
4	Chao Luo, Xiaoqiang Liu, Kai Zhang, Qinghong Chang. "A Recommendation System for Cloud Services Based on Knowledge Graph", 2018 IEEE 9th International Conference on Software Engineering and Service Science (ICSESS), 2018 Publication	<1%
5	P. Shanmughasundaram, S Thirumurugaveerakumar, R Kalaiselvi.	<1%

	"Chapter 28 Influence of Turning Parameters on the Surface Roughness and Cutting Force of the Aluminum Matrix Hybrid Composites", Springer Science and Business Media LLC, 2023 Publication	
6	export.arxiv.org Internet Source	<1 %
7	Muhammad Usman Hadi, qasem al tashi, Rizwan Qureshi, Abbas Shah et al. "Large Language Models: A Comprehensive Survey of its Applications, Challenges, Limitations, and Future Prospects", Institute of Electrical and Electronics Engineers (IEEE), 2023 Publication	<1 %
8	deepai.org Internet Source	<1 %
9	www.catalyzex.com Internet Source	<1 %
10	Kong, Dehua, and Yuqing Zhai. "Trust Based Recommendation System in Service-oriented Cloud Computing", 2012 International Conference on Cloud and Service Computing, 2012. Publication	<1 %
11	coek.info Internet Source	<1 %

12	repository.nii.ac.jp Internet Source	<1 %
13	www.hindawi.com Internet Source	<1 %
14	Daixing Zhong, Gangjun Yang, Jiashuang Fan, Baozhen Tian, Yukun Zhang. "A service recommendation system based on rough multidimensional matrix in cloud-based environment", Computer Standards & Interfaces, 2022 Publication	<1 %
15	Peilin Zhou, Jingqi Gao, Yueqi Xie, Qichen Ye, Yining Hua, Jaeboum Kim, Shoujin Wang, Sunghun Kim. "Equivariant Contrastive Learning for Sequential Recommendation", Proceedings of the 17th ACM Conference on Recommender Systems, 2023 Publication	<1 %
16	"Operationalizing Multi-Cloud Environments", Springer Science and Business Media LLC, 2022 Publication	<1 %

Exclude quotes	On	Exclude matches	Off
Exclude bibliography	On		