


Article

Exploring the Potential of Large Language Models in Radiological Imaging Systems: Improving User Interface Design and Functional Capabilities

Luyao Zhang ¹, Jianhua Shu ¹, Jili Hu ¹, Fangfang Li ¹, Junjun He ², Peng Wang ^{3,*} and Yiqing Shen ^{4,*} ¹ School of Medical Informatics Engineering, Anhui University of Chinese Medicine, Hefei 230031, China² Shanghai AI Laboratory, Shanghai 200233, China³ Graduate School, Anhui University of Chinese Medicine, Hefei 230031, China⁴ Department of Computer Science, Johns Hopkins University, Baltimore, MD 21218, USA

* Correspondence: anhuiwangpeng@126.com (P.W.); yshen92@jhu.edu (Y.S.)

Abstract: Large language models (LLMs) have demonstrated remarkable capabilities in natural language processing tasks, including conversation, in-context learning, reasoning, and code generation. This paper explores the potential application of LLMs in radiological information systems (RIS) and assesses the impact of integrating LLMs on RIS development and human–computer interaction. We present ChatUI-RIS, a prototype chat-based user interface that leverages LLM capabilities to enhance RIS functionality and user experience. Through an exploratory study involving 26 medical students, we investigate the efficacy of natural language dialogue for learning and operating RIS. Our findings suggest that LLM integration via a chat interface can significantly improve operational efficiency, reduce learning time, and facilitate rapid expansion of RIS capabilities. By interacting with ChatUI-RIS using natural language instructions, medical students can access and retrieve radiology information in a conversational manner. The LLM-powered chat interface not only streamlines user interactions, but also enables more intuitive and efficient navigation of complex RIS functionalities. Furthermore, the natural language processing capabilities of LLMs can be harnessed to automatically generate code snippets and database queries, accelerating RIS development and customization. Preliminary observations indicate that integrating LLMs in RIS has the potential to revolutionize user interface design, enhance system capabilities, and ultimately improve the overall user experience for radiologists and medical professionals.

Keywords: large language model (LLM); radiological information system (RIS); human–computer interaction



Citation: Zhang, L.; Shu, J.; Hu, J.; Li, F.; He, J.; Wang, P.; Shen, Y. Exploring the Potential of Large Language Models in Radiological Imaging Systems: Improving User Interface Design and Functional Capabilities. *Electronics* **2024**, *13*, 2002. <https://doi.org/10.3390/electronics13112002>

Academic Editor: Dimitris Apostolou

Received: 23 April 2024

Revised: 17 May 2024

Accepted: 20 May 2024

Published: 21 May 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Human–computer interaction (HCI) plays a crucial role in determining the usability and reliability of various information systems in hospitals [1]. The development of intuitive graphical user interfaces (GUIs) has led to significant advancements in HCI, resulting in more efficient and user-friendly software interfaces [2,3], such as streamlined navigation, optimized layout, and enhanced feedback mechanisms. Existing Radiological Information Systems (RIS) rely on GUIs as the primary interface for users. However, despite efforts to tailor interfaces of RIS to specific needs, challenges related to usability and workflow efficiency persist [4,5]. Specifically, radiologists often face cumbersome navigation processes, requiring multiple clicks and significant time to access pertinent information [6]. These inefficiencies lead to workflow inconsistencies and hinder radiologists' productivity. Furthermore, the complex operating logic in RIS interfaces results in intricate function logic, posing challenges for both experienced professionals and trainees [7,8]. The steep learning curve associated with understanding operational procedures and system component logic underscores the need for optimizing HCI in RIS to enhance user experience and efficiency.

The emergence of large language models (LLMs) has provided new solutions for HCI [9], with natural language-based interfaces becoming increasingly familiar to users [10]. LLMs exhibit impressive capabilities in tasks such as question answering, text generation, summarization, and human language understanding [11–14]. Moreover, LLMs demonstrate proficiency in reasoning and planning tasks [15,16], comprehending task objectives, refining tasks, and enabling automatic execution through API calls [17–20]. In the radiology domain, LLMs have been employed to interpret radiology reports [21], and further research has explored infusing LLM prompts with UI design in the prototyping process to reduce the creation time of functional prototypes [20]. Motivated by these advancements, we propose leveraging LLMs in the user interface of RIS to enhance the interaction. Our vision is to harness the understanding, planning, and coding capabilities of LLMs to address the challenges associated with learning and operating RIS. To be more specific, this work explores the following research questions:

- How can the integration of LLMs facilitate RIS function operation?
- In what ways can LLMs assist trainees in effectively learning RIS functional logic?
- How can an LLM-integrated user interface help trainees operate RIS more easily?
- Can LLMs extend the capabilities of RIS beyond their current scope?

To address these questions, we conducted an exploratory study, and found that:

1. A chat-based user interface (ChatUI) could successfully complete three core RIS function operations.
2. Infusing LLMs into RIS has the potential to reduce learning costs and increase learning efficiency for trainees.
3. Combining ChatUI with LLMs may help trainees reduce operation times and minimize failure rates.
4. LLMs can expand the statistical analysis capabilities of RIS by leveraging their generating, summarizing, and planning abilities.

2. Related Works

2.1. Radiological Information Systems

RISs are crucial components of modern healthcare infrastructures, facilitating the management of radiological images, patient data, and workflow processes. With the increasing popularity of computers, RISs designed with GUIs has become more familiar to patients and physicians and are still widely used today. Several studies have investigated the usability and user experience of RISs [22–24]. Inadequate workflow support, inefficient system operation, and insufficient user-centered interface design have been identified as factors contributing to poor user experience in RISs [24]. In particular, poor interface design elements such as disappearing control buttons, the need to memorize menu and button names and commands, and repeated operations to perform routine tasks have shown a strong correlation with suboptimal user experiences [23]. The integration of radiology and hospital information systems has led to the concentration of massive patient data from other systems into RISs in real-world scenarios, resulting in more complex workflow optimization [25]. This complexity is reflected in user interfaces (UIs) that incorporate multi-operation pages and numerous UI elements [24]. These intricate user interfaces have increased staff workload and decreased work efficiency [26]. Recent research has explored the integration of advanced technologies, such as AI and NLP, to improve HCI in RIS systems [27–30].

2.2. Large Language Models

Large Language Models (LLMs) are advanced AI models designed to understand and generate human-like text [31]. A well-known example of an LLM is the Generative Pre-trained Transformer (GPT), developed by OpenAI [32]. GPT models are capable of performing a wide range of language tasks, including text generation [12], translation [33], summarization [14], and question answering [34]. In the healthcare domain, LLMs have been used for various tasks, such as summarizing reports [21], providing medical ad-

vice [35], analyzing mental health [36], and answering medical questions [37]. These studies have demonstrated the great potential of LLMs in handling healthcare-related tasks.

For instance, RaDialog [38], an interactive domain-specific large vision-language model, pairs LLMs with radiology images to generate comprehensive radiology reports and facilitate real-time conversations with radiologists. RaDialog's capabilities in interactive tasks, such as report correction, summary generation, discovery prediction, and answering questions, represent foundational steps toward a clinical dialogue system for LLMs. Effective and timely human–computer interaction is crucial for the efficient operation of healthcare systems. LLMs, with their strong conversational abilities, show immense potential in enhancing HCI within RIS systems.

2.3. Natural Language-Based Human–Computer Interaction with LLMs

A common feature of LLMs is the use of natural language to communicate with humans, as exemplified by ChatGPT. This type of natural language-based human–computer communication represents a new form of HCI [39]. Researchers have explored this new HCI paradigm using LLMs in several domains. For example, PromptInfuser [20] demonstrates the possibility of using LLMs that incorporate the power of prompts to create functional UI mock-ups. TaskMatrix.AI [40] explores the feasibility of LLMs autonomously handling complex tasks in the context of natural language dialogue. These studies highlight the great potential of LLMs for HCI. However, the application of LLMs as user interfaces in RIS systems has not been thoroughly investigated in previous works.

3. Methods

In this section, we elaborate on the details of a ChatUI-RIS system based on LLM as the interface, with a user interaction page designed using a chatbox element and a system function workflow controlled by the LLM. The LLM calls API interfaces to process user instructions. As shown in Figure 1, the overall architecture of the ChatUI-RIS system consists of the following three components:

1. LLM as Interface: the LLM serves as a unified interaction interface for communicating with users, understanding their context, and presenting results based on the LLM's processing.
2. LLM as Controller: the LLM is responsible for designing RIS system-specific task plans, selecting relevant APIs, and generating the corresponding API calling code.
3. LLM as Executor: the LLM follows the specific task plan, executes the generated action codes, and returns the final execution results.

The LLM-based interface component provides a natural language communication channel between the user and the RIS system. By leveraging the LLM's language understanding capabilities, the system can interpret user queries, instructions, and context to provide appropriate responses and guide the user through the interaction process. The controller component utilizes the LLM's reasoning and planning abilities to design task-specific workflows within the RIS system, which can analyze user requirements, identify the necessary system functions, and select the appropriate APIs to accomplish the desired tasks. The LLM then generates the code required to call the selected APIs, enabling the system to perform the requested actions. Finally, the executor component takes the task plan and generated code from the controller and executes the actions by invoking the appropriate API calls. It manages the execution process, handles any intermediate results or errors, and returns the final output to the user through the LLM-based interface.

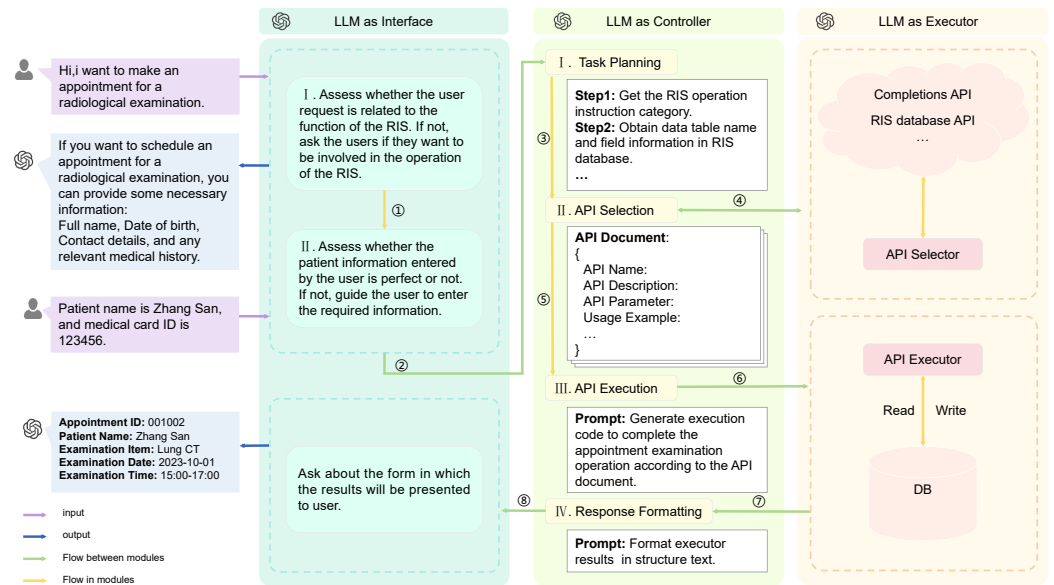


Figure 1. An overview of the proposed ChatUI-RIS. Firstly, the LLM as an Interface component interacts with users through natural language, understanding their requests and presenting results. Then, the LLM as Controller assesses the user’s request, designs task plans, selects relevant APIs, and generates API calling code. Finally, the LLM as Executor follows the task plan, executes the generated code, and returns the final results to the user via the interface. The system leverages the LLM’s language understanding, reasoning, and execution capabilities to provide a conversational, user-friendly experience for interacting with the RIS.

3.1. LLM as Interface

The LLM as Interface component serves as the human–machine interface of our RIS, enabling users to interact with the system through natural language conversations. This interactive mode can respond promptly and clearly when user input is ambiguous or out of scope, providing a user-friendly experience. The LLM interface component has two main capabilities. First, it performs task categorization and information assessment by taking user instructions and conversational context as input to understand the user’s intention and assess whether the instruction falls within the RIS’s functionality. Users may potentially provide instructions that are unrelated or outside the system’s scope; the LLM as Interface component leverages its understanding of the conversational context to categorize these instructions. If the task aligns with the RIS’s capabilities, the LLM proceeds to the next step; otherwise, it replies to the user based on its role setting and guides them to provide a valid instruction. The LLM then parses the user’s instruction again and estimates the adequacy of the information provided for task execution. If the user’s input satisfies the task completion requirements, the LLM interface sends the parsed task to the subsequent LLM as Controller component. If additional information is needed, the LLM interface prompts the user and provides examples to clarify the requirements. Second, the LLM interface presents the execution results to the user in a format suitable for the specific task type. As different tasks have non-uniform output formats, the LLM interface reformats the execution result according to preset task settings, ensuring that the results are clear and easily understandable for the user.

By leveraging its natural language understanding capabilities, the LLM as Interface component interprets user instructions, assesses task categories, and determines the necessary information for task execution. This allows users to interact with the system using natural language, even if their initial instructions are ambiguous or incomplete. The LLM as Interface component guides users by requesting additional information and providing examples when needed, ensuring that tasks can be accurately understood and processed by the system. Moreover, by adapting the output format based on preset task settings, the LLM as Interface component ensures that the results are easily comprehensible and actionable

for the user. This tailored presentation of results enhances the overall user experience and facilitates effective communication between the user and the RIS.

3.2. LLM as Controller

The LLM controller component is the core of our proposed ChatUI-RIS, responsible for controlling the execution flow of each task. The LLM controller leverages its generating, reasoning, and planning capabilities to create task plans, select the most suitable APIs, generate API execution code, and format execution results in structured text. The controller consists of four main parts:

- **Task planning:** The LLM receives user inputs from the LLM as Interface component and plans the task completion steps and execution. These steps include recognizing the task instruction category, assessing the integrity of key data required for task execution, deciding when and how to call APIs, inferring SQL operations based on the task instruction category, and analyzing access to databases and tables using key task data.
- **API selection:** When the task plan indicates the need to call an API, the LLM controller sends an API calling request to the subsequent LLM as Executor component. The Executor identifies the specific task APIs and sends back API-calling documents to the LLM as Controller component, which contains essential information including the API's name, function description, input and output parameters, and usage examples. The LLM controller learns from these documents to select the most appropriate APIs for each task.
- **API execution:** LLMs possess a powerful code-generation capability, which is utilized by the controller to generate executable action code that satisfies the task instructions according to the API documents. The generated action code is then sent to the LLM as Executor component for API execution. This process ensures that the selected APIs are properly invoked and executed to accomplish the desired task.
- **Response formatting:** The raw execution results may not be directly usable or easily understandable by the user. To address this, the LLM controller employs its in-context understanding ability to transform the structured text output into human-readable text. This formatted response is then sent back to the LLM as Interface component for presentation to the user. By converting the results into a more comprehensible format, the LLM as Controller component enhances the user experience and facilitates effective communication of the task outcomes.

The task planning phase is crucial for determining the necessary steps and data required for successful task execution. The LLM as Controller component analyzes user inputs, recognizes task categories, assesses data integrity, and makes informed decisions on API invocation and database access. During the API selection phase, the LLM as Controller component collaborates with the subsequent LLM as Executor component to identify and learn about the specific APIs needed for each task. By receiving detailed API documents, the LLM as Controller component can make informed decisions on which APIs to use, ensuring that the most appropriate and efficient APIs are selected for each task. The API execution phase capitalizes on the LLM's code-generation capabilities to produce executable action code based on the selected APIs and task instructions. This generated code is then sent to the LLM as Executor component for execution, ensuring that the APIs are properly invoked and the desired task outcomes are achieved. Finally, the response formatting phase utilizes the LLM's in-context understanding ability to transform the structured execution results into human-readable text, which enhances the user experience by presenting the task outcomes in a clear and easily comprehensible manner.

3.3. LLM as Executor

This component is responsible for retrieving APIs and executing API calling code to interface with the RIS system database and other third-party platforms. It contains two main functions:

- **API selector:** The API selector parses the API calling request from the LLM as Controller component and identifies suitable APIs based on the available API library. It leverages the LLM's understanding and modular strategy to retrieve task-related APIs that align with the calling requirements and task planning. The retrieved API documents, which include essential information such as the API's name, function description, input and output parameters, and usage examples, are then sent back to the Controller component. This process ensures that the most appropriate APIs are selected for each task, enabling efficient and accurate execution.
- **API executor:** The API executor receives the action code generated by the LLM as Controller component and executes it to run the selected APIs. The APIs can range from simple HTTP requests to more complex operations such as Text-to-SQL algorithms or ML or DL models that require multiple input parameters. The API executor ensures that the generated code is properly executed, interfacing with the RIS database and other third-party platforms as necessary. This execution process is crucial for retrieving data, performing calculations, and carrying out the desired tasks based on the user's instructions and the task planning.

After completing the API execution, the LLM as Executor component provides feedback on the execution results to the previous LLM as Controller component. The results are delivered in a structured text format, which requires further processing by the LLM as Controller component to transform them into human-readable responses.

4. User Study

4.1. User Study Overview

We conducted an exploratory user study to verify our three hypotheses.

Hypothesis 1. *An LLM, when used as a human–machine interface, can effectively support RIS operations.*

Hypothesis 2. *A natural language chat interface can help users understand RIS function logic more easily.*

Hypothesis 3. *A natural language chat interface can enable users to operate RIS functions more efficiently.*

The user study consisted of five parts:

1. **Function selection:** We chose three major functions in the RIS system (examination appointment, report retrieval, and data analysis) as the focus of the user study. We created all the necessary test data and stored it in a local database. The test data consisted of sample case data used in medical teaching, rather than real clinical data. Test data for the user study includes patients' demographic information and diagnosis information, where we only choose related columns for the user study, for example: patient ID, gender, age, inspection item, examination date, and so on. The parts of test data are shown as Table 1.
2. **Demo program development:** we developed a demo program that included the three main RIS system functions, with two human–machine interface links.
3. **Experiment design:** We designed the experiment to include three tasks, each corresponding to one of the functions in the demo program. We also created task test questionnaires to collect user test data.
4. **Experiment implementation:** the formal experiment involved recruiting participants, preparing and conducting the experiment, and collecting questionnaires.
5. **Data analysis:** finally, we analyzed the collected questionnaires to verify whether our hypotheses held true.

Table 1. The parts test data of demo program.

PatientID	Gender	Age	Inspection Item	Examination Date	Diagnosis	...
20110153729	M	37	X-ray	22 May 2019	clavicle fracture	...
20192844100	F	19	CT	7 January 2020	pneumonia	...
17934006876	F	51	X-ray	14 August 2021	pulmonary tuberculosis	...
20114596237	M	48	X-ray	5 November 2020	radius	...
20486229815	M	42	Ultrasound	23 March 2021	uterine leiomyomatosis	...
...

4.2. Prototype Implementation

We chose examination appointment, report retrieval, and data analysis, which are the main modules of the RIS, as the focus of our prototype implementation. The prototype integrates both a GUI and a ChatUI for comparison. The GUI interface consists of three panels for each functionality, designed to resemble a real RIS. Extra buttons were removed, and only necessary components were retained, so as to streamline the user experience in GUI. Figure 2a–c illustrates the GUI interface of our prototype implemented for the user study. In contrast, the ChatUI interface features all functions with a unified interface, with a design inspired by ChatGPT (<https://openai.com/blog/chatgpt/> (accessed on 1 May 2024)).

The interface includes a dialogue area, a text input box, and a submit button, as shown in Figure 2d. The prototype was developed using Python, 3.8.13 and the front-end pages were created using the Gradio (<https://www.gradio.app/> (accessed on 1 May 2024)) tool.

Here is an example to illustrate how interaction through GUI interface and ChatUI interface, as shown in Figure 3.

To ensure the ChatUI interface information is true and valid enough, we design prompts to help our system understand user requests clearly and respond based on facts. When the user visits the ChatUI interface, the chatbot shows RIS function task entry points with labels. When the user chooses one of the RIS tasks, the chatbot will prompt the user and provide examples to clarify the requirements. If the user does not choose the task label and types their request directly, the chatbot will assess first whether user instructions fall within the RIS's functionality. If the user has unrelated instructions or is outside the system's scope, the chatbot will deny the user's request by friendly reply. The chatbot will parse the user's instructions again and estimate the adequacy of information provided by the user further. It will continue prompting the user to type more information until the chatbot collects enough data to execute the task. Before providing chatbot feedback on execution results, we added a strict answer-verify prompt to constrain the chatbot to only generate result content from data retrieved from the RIS database and refuse to make up data. If there is no data retrieved, the chatbot will truthfully reply with no relevant content. In that case, the chatbot only replies with task execution results based on the data acquired from the local database, reducing the occurrence of nonsense.

Radiological Imaging Systems(GUI demo)

Appointment Page Retrive Page Analyses Page

Application ID *	Patient name *	Ward Name	Bed Number
Identification ID *	Patient Age	Gender	Marital Status
Diagnosis *	Inspection items *	Billing Doctor	Address

Applyate

(a)

Radiological Imaging Systems(GUI demo)

Appointment Page Retrive Page Analyses Page

Examination ID:	Patient name:	Search

Report List

Examination ID	Patient name	Gender	Age	Examination Term	Examination Date

(b)

Radiological Imaging Systems(GUI demo)

Appointment Page Retrive Page Analyses Page

Data Statistics

By Item By Doctor By Ward

Statistic results

Items	Counts

Data Analysis

Disease	Relevant factor	Analysis

(c)

Radiological Imaging Systems (Chat-UI demo)

Enter your information below, separated by commas and ended by a period.

Chatbot

Please Enter RIS Operation information. **Submit**

[Examples](#)

Examination Appointment Report Retrive Data Analysis

(d)

Figure 2. User interfaces of the prototype implemented for our user study. (a) The examination tab of the GUI interface is designed to resemble a real RIS, with streamlined components and functionality. (b) The report retrieves the tab of the GUI interface. (c) The data analysis tab of the GUI interface. (d) The ChatUI interface, inspired by ChatGPT, features a dialogue area, text input box, and submit button for natural language interaction with the system.

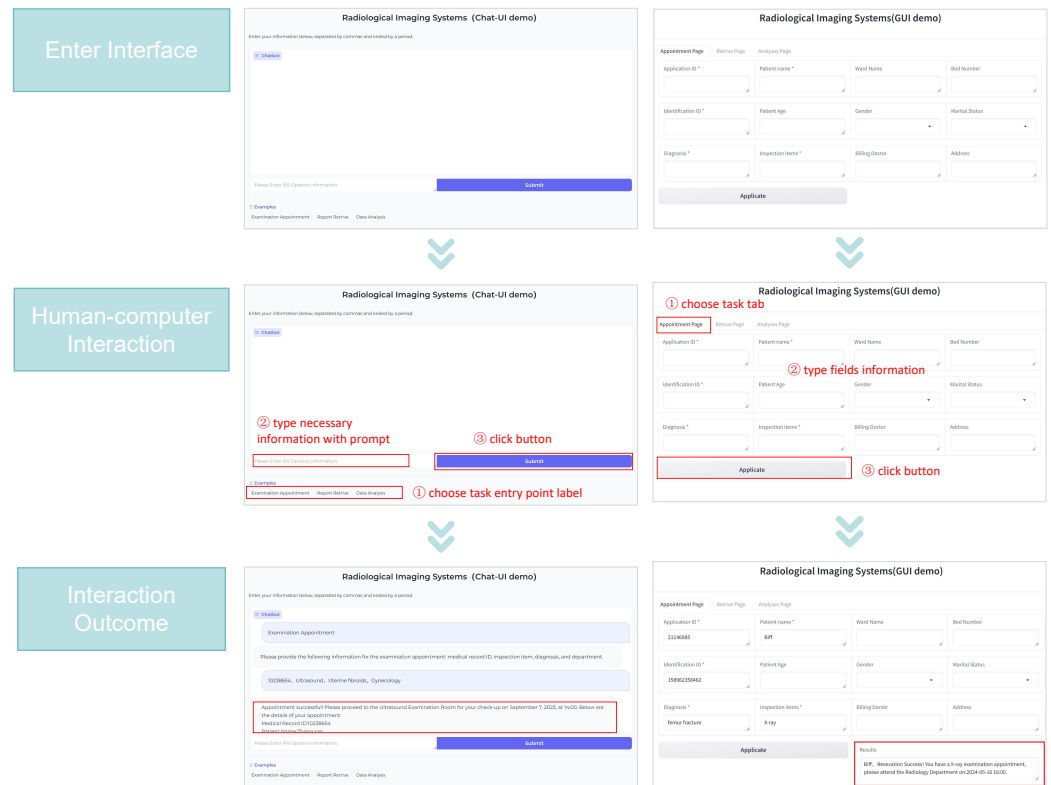


Figure 3. The interaction through the GUI interface and ChatUI interface. For the GUI interface, first, find the task tab of the demo when getting the task; then, type the input fields according to input item labels and click the submit button; finally, the GUI interface feedback on the task operation result with a pop-up window. For the ChatUI interface, first, type instructions in the input box after getting the task; then, type the necessary information according to the chatbot prompt; finally, the GUI interface feedback on the task operation results in a reply.

4.3. Questionnaire Design

To comprehensively evaluate the two interaction methods (GUI and ChatUI) in our prototype, we designed a questionnaire focusing on three key aspects: functional integrity, learning difficulty, and ease of operation. The functional integrity component of the questionnaire assesses the effectiveness of the two interaction methods in implementing the core functionalities of the RIS system. This evaluation aims to test our first hypothesis (H1), which posits that natural language dialogue interaction can successfully support the operation of an RIS. By comparing the functional performance of the GUI and ChatUI interfaces, we can determine the extent to which the ChatUI method can fulfill the essential functions of the RIS. The learning difficulty aspect of the questionnaire investigates the differences in the learning process between the two interaction modes. This assessment is designed to test our second hypothesis (H2), which suggests that a natural language chat interface can facilitate a more intuitive understanding of the RIS’s functionality. By comparing users’ learning experiences with the GUI and ChatUI interfaces, we can evaluate the potential of the ChatUI method to reduce the learning curve associated with RIS operation. Finally, the ease of operation component of the questionnaire examines the perceived difficulty of using the two interaction modes and compares their relative popularity among users. This evaluation tests our third hypothesis (H3), which proposes that a natural language chat interface can enhance the efficiency of RIS function operation.

4.4. User Recruitment

Given the highly specialized nature of the RIS and the specific requirements for its operators, we established a set of criteria for recruiting participants to ensure the fairness and validity of the experiment. The recruitment criteria were as follows:

- Participants must have a background in the medical field, including those who have worked in a hospital's imaging department, whose research focuses on the medical domain, or whose specialization is in a medical-related field.
- Participants should not be experts in the field of medical imaging.
- Participants should not be staff members of a hospital's imaging department with more than one year of working experience in the field.

The rationale behind excluding field experts and experienced staff was to mitigate the potential confounding effects of their familiarity with RIS systems on the experimental results. Participants with extensive prior knowledge or experience could introduce bias and hinder the accurate assessment of the learning curve and usability of the GUI and ChatUI interfaces for novice users. Following the establishment of the recruitment criteria, we successfully recruited 26 participants for the study. To ensure the suitability of the recruited participants, we conducted a background survey to gather information about their medical field experience, expertise in medical imaging, and familiarity with RIS. This background information was used to verify that the participants met the recruitment criteria and to provide context for interpreting the experimental results.

4.5. User Study Workflow

Before formally conducting the experiment, it is crucial to provide participants with an introduction to the experimental procedures and familiarize them with the system. We conducted user training sessions to present a detailed overview of the experiment flow, which consists of six steps (Figure 4).

1. Introduction to the experiment background and objective: We introduced the participants to the experiment's background and the main objective, which is to determine whether the ChatUI interaction, empowered by large language models, can offer a more intelligent and efficient solution for human–computer interaction tasks compared to the traditional GUI interaction. By providing users with a clear understanding of the purpose of the experiment, they can better compare the two interaction methods.
2. Overview of the experiment content: The experiment consists of three tasks: (1) a radiological examination appointment test, (2) a radiological report retrieval test, and (3) a data statistical analysis test. Participants were informed about the nature and scope of each task.
3. User background survey: We conducted a background survey to collect information about the participants' experience and expertise. This survey helps us better understand if the participants' backgrounds influence their interaction with the system.
4. Random assignment: We designed two experimental conditions to avoid any potential bias due to the order of interaction with the two systems. In the experimental group (ChatUI interaction), participants used the ChatUI system, while in the control group (traditional GUI interaction), participants used the GUI system. The conditions were assigned randomly to the participants:
 - Condition 1: a subset of participants performed the experimental group test first, followed by the control group test.
 - Condition 2: a subset of participants performed the control group test first, followed by the experimental group test.
5. Experiment execution: after assigning the groups and operating conditions, the participants formally conducted the experiment.
6. Experiment feedback: upon completing each group test, participants filled out a feedback survey to record their experiences and impressions of the interaction method.

After finishing all group tasks, participants completed a user experience feedback questionnaire to document and score their experiences and feelings regarding the two interaction methods. In addition to introducing experiment flow, we also informed partici-

pants how to deal with operation error. They were permitted to look up guides during the experiment. We recorded the flipping times and task operation times.

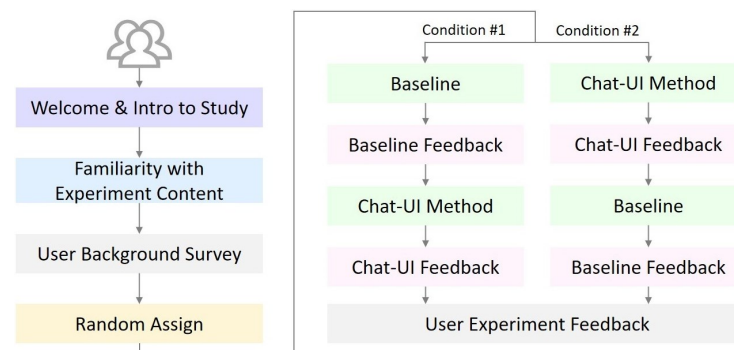


Figure 4. The six-step user study workflow.

5. Experiment Result

We present four studies to demonstrate the successful application of our approach. The first study investigates whether an LLM-based human–machine interface can effectively support RIS operations. The second study explores whether a natural language chat interface can facilitate an easier understanding of RIS function logic. The third study examines whether a natural language chat interface can enhance the efficiency of RIS function operation. Finally, the fourth study assesses the capability of the chat interface method to perform additional ML data analysis functions.

5.1. Study 1: Feasibility of LLM-Based Chat Interaction in RIS

This experiment aims to demonstrate that users can directly operate the RIS through a chat interface, addressing the question of whether chat interaction can be successfully employed in an RIS.

5.1.1. Study Design

The experiment involved 26 subjects divided into two groups. Group 1 tested the GUI interface first, followed by the ChatUI interface. Conversely, Group 2 tested the ChatUI interface first, then the GUI interface. Each subject tested a total of six functions, three with GUI and another three with ChatUI. All subjects used identical experimental conditions and machines for the testing and evaluation sessions. The experimental manipulations were introduced during the evaluation session, and the questionnaire was administered under the same computer conditions as the testing and evaluation sessions. Subjects tested three functions, respectively.

5.1.2. Study Results

The results showed that, despite the chat interface being built simply based on an LLM model, subjects were able to accurately operate the RIS system after brief training. As predicted, when assessing the Chat HCI performance under the same computer conditions, most subjects successfully completed RIS system tasks via the chat box (Figure 5). These findings support our hypothesis that users can directly operate the RIS system through a chat interface.

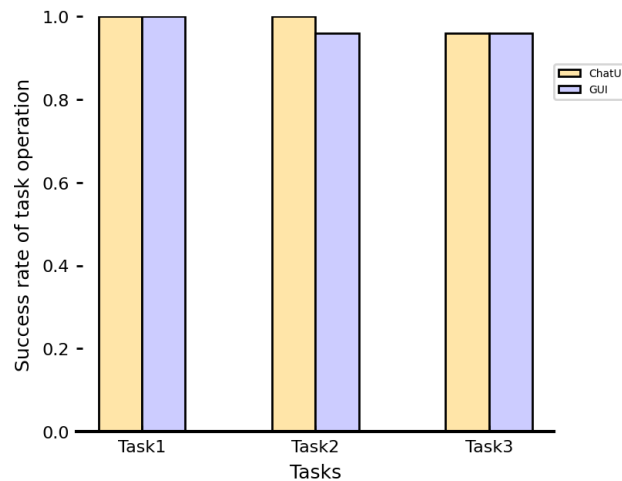


Figure 5. Success rate of task operation using ChatUI and GUI interfaces. The graph compares the performance of ChatUI and GUI for three different tasks. The success rate is consistently high for both interfaces across all tasks, demonstrating the feasibility of using ChatUI for RIS operations.

5.2. Study 2: Chat Interface for Easier Understanding of RIS Function Logic

This experiment investigates whether a natural language chat interface can help users understand RIS function logic more easily. We aim to demonstrate that users can learn RIS function logic more effectively through agent prompts built based on LLMs.

5.2.1. Study Design

In this experiment, the 26 subjects were allocated to the same groups as in Study 1. Before the experiment, subjects were pooled together to receive a brief guideline introduction about the demo program functions. All subjects used the same experimental conditions and machines for the testing and evaluation sessions. During the test session, subjects were tested on three tasks using both the ChatUI interface and GUI interface. Each subject needed to remember the number of times they looked up the guidelines (Table 2, indicator 2) while operating. Indicator 2 was recorded in the questionnaire for assessing the testing. During the evaluation session, subjects evaluated the learning experience of both interfaces based on their manipulation experience (Table 2, indicator 3).

Table 2. The main indicators of 4 studies.

Study Number	Index
1	indicator 1: Task success or not
2	indicator 2: Look up guideline indicator 3: Learning hard degree
3	indicator 4: Operation times indicator 5: Operation failure times indicator 6: Operation time
4	indicator 7: Data analysis

5.2.2. Study Results

The experiment revealed that, in the “Chat interface” condition, most subjects (17 out of 26) could complete the tasks without referring to the guidelines. In contrast, in the “GUI interface” condition, only a few subjects (10 out of 26) completed the tasks without looking up the guidelines. Furthermore, the “GUI interface” condition had a higher percentage of subjects who flipped the guidelines two or more times compared to the “Chat interface” condition. The agent prompt could correctly guide subjects through tasks step-by-step, reducing the need to memorize all functional logic and thereby shortening learning time.

To analyze the subjects’ learning efficiency in both conditions, we conducted a statistical analysis of the Study 2 questionnaire. The results of analyzing indicator 3 showed that 88.5% of subjects found the method of prompting function logic in a chat format easier to learn, while 80.8% of subjects still considered the window-button approach relatively acceptable (Figure 6). Hence, a simple UI and accurate prompts can help users better understand task workflows and sequential logic, facilitating easier learning progress.

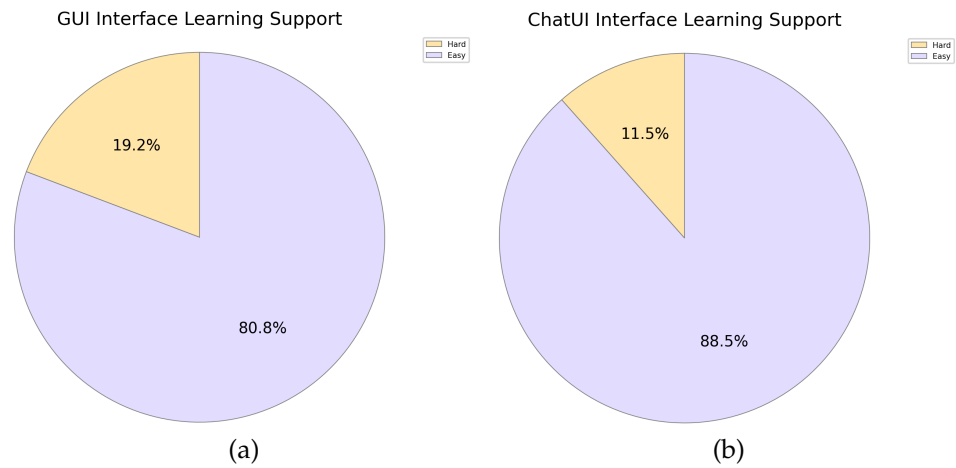


Figure 6. User evaluation of learning support for GUI and ChatUI interfaces. The majority of subjects (88.5%) found the ChatUI interface easier to learn compared to the GUI interface (80.8%), despite most subjects using ChatUI for the first time. This suggests that the natural language prompts in ChatUI facilitate a better understanding of RIS function logic. (a) Evaluation of the GUI interface learning support. (b) Evaluation of the ChatUI interface learning support.

When subjects were accustomed to the GUI operation interface, they demonstrated a higher support rate for the Chat interface, despite most subjects using it for the first time. To further investigate the learning process, we analyzed the number of times subjects looked up guidelines for both interfaces. The Chat interface, which draws users’ attention to learning function logic through agent prompting, had a lower learning difficulty compared to the GUI approach (Figure 7). These findings confirm our hypothesis that the chat interface can help users understand RIS function logic more easily.

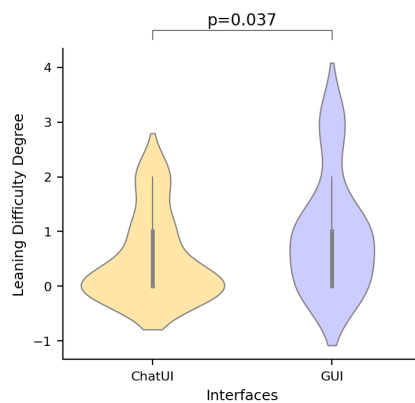


Figure 7. Comparison of learning difficulty between ChatUI and GUI interfaces. The ChatUI interface, which guides users to learn function logic through agent prompting, demonstrates a significantly lower learning difficulty compared to the GUI approach. This result supports the hypothesis that the chat interface can facilitate an easier understanding of RIS function logic.

5.3. Study 3: Chat Interface for More Efficient RIS Function Operation

This experiment investigates whether a natural language chat interface can help users operate RIS functions more efficiently. We aim to demonstrate that users can complete RIS tasks more easily through a chat approach. The system operation difficulty tested in this experiment is the same as in Study 2; however, this study further examines operation efficiency.

5.3.1. Study Design

In this experiment, 26 subjects participated. The experimental setup was identical to Study 2, except that subjects needed to remember the number of operations and error times during the testing session. When completing a task, the experiment instructor recorded the task operation time, and subjects recorded these indicators in the questionnaire (Table 2, indicators 4–6).

5.3.2. Study Results

We found that most subjects could finish the three tasks with a single operation. In the “Chat interface” condition, 6 subjects required two or more operations, while in the “GUI interface” condition, 11 subjects needed multiple operations. This suggests that operating the GUI interface required more attempts than operating the Chat interface on the same computer and with the same guideline introduction. Consequently, subjects encountered more error cases when using the GUI interface. To complete three tasks, the “GUI interface” total number of operation times was higher than the “Chat interface”. In other words, the extra operation times that occurred in “GUI interface” caused by more error operations. We then analyzed the operation efficiency indicators. For the “operation difficulty” indicator, 84.6% of subjects found the ChatUI interface more convenient to operate, while 69.2% of subjects said they were more proficient in operating the GUI interface (Figure 8). Notably, subjects preferred a concise system interface. For multiple operations in both interfaces, the GUI interface had a significantly higher operation error rate compared to the Chat interface (Figure 9). This implies that a system interface with multiple and varied UI elements increases the probability of users clicking on incorrect buttons or interacting with unintended UI elements. Instead, a simplified UI can help users focus more on the task workflow and the completeness of its functions, reduce the probability of incorrect operations leading to task failure, and improve overall efficiency. These results are consistent with the hypothesis that a chat interface makes it easier for users to operate RIS functions.

Despite the higher convenience and lower error rates associated with the Chat interface, we found that subjects spent more time operating the Chat interface compared to the GUI interface (Figure 10). This result can be attributed to the fact that longer accessing time to OpenAI API, and network latency caused an increase in operation time. It is important to note that this longer operation time is not inherently related to the usability or efficiency of the Chat interface itself, but rather a consequence of the specific network conditions during the experiment.

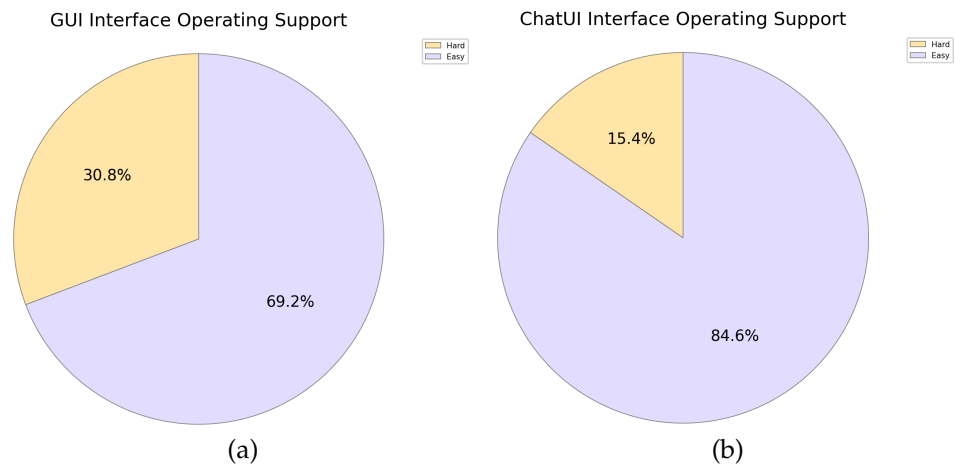


Figure 8. User evaluation of operation support for GUI and ChatUI interfaces. The majority of subjects (84.6%) found the ChatUI interface more convenient to operate, while 69.2% of subjects reported being more proficient in operating the GUI interface. These results suggest that the ChatUI interface provides a more user-friendly experience, despite users’ familiarity with traditional GUI interfaces. (a) Evaluation of the GUI interface operating support. (b) Evaluation of the ChatUI operating support.

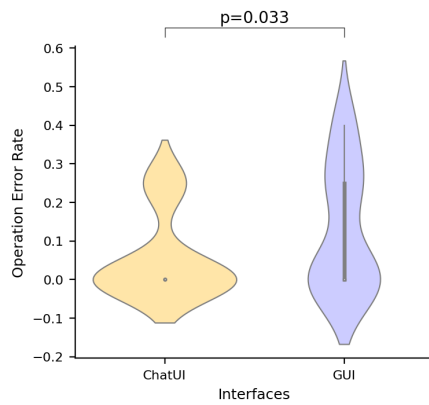


Figure 9. Comparison of operation error rates between GUI and ChatUI interfaces. The GUI interface exhibits a significantly higher operation error rate compared to the ChatUI interface when multiple operations are required.

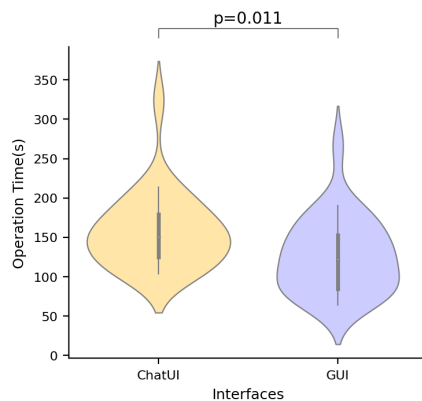


Figure 10. Comparison of operation times between GUI and ChatUI interfaces. Subjects spent more time operating the ChatUI interface compared to the GUI interface. However, this longer operation time is attributed to network latency caused by accessing OpenAI’s API, rather than inherent usability or efficiency issues with the ChatUI interface itself. The operation time difference is a consequence of the specific network conditions during the experiment.

5.4. Study 4: Chat Interface for Expedient Function Expansion

This experiment investigates whether a chat interface supports expedient function expansion, specifically whether LLMs can support RIS function expansion by calling APIs. The functions tested in this experiment are the same as those in Study 1; however, this study further examines LLMs' ability to extend functionality in the data analysis module of RIS.

5.4.1. Study Design

This experiment is an extension of the previous studies. As the RIS used in the earlier studies did not have an analysis module, we added a data analysis module to the prototype for this function expansion test. In the chat interface, the module utilized machine learning models by calling related APIs, and the results were presented with both text and charts. In the GUI interface, the module employed traditional statistical learning algorithms, and the operation results were presented only with charts. In this experiment, 20 subjects were divided into two groups. As in Study 1, the two groups tested the data analysis module in the Chat interface and GUI interface, respectively. We provided a set of heart disease data for testing. During the evaluation session, subjects needed to answer questions based on the results they obtained, indicating which approach allowed them to directly understand the data analysis conclusions in the data analysis module.

5.4.2. Study Results

In this experiment, all 20 subjects successfully completed the test via the chat interface, while 19 subjects completed the test via the GUI interface. The results are consistent with the chat interface supporting expedient function expansion through calling APIs in RIS. When assessing the friendliness of the two interfaces for function expansion, subjects responded differently. When the same chart format was presented, subjects could not discern the difference between the two interfaces. However, when the chat interface presented a conclusion text alongside the chart, subjects found the operation results more intuitive. Specifically, for the data analysis task, subjects preferred the approach that presented results with both charts and text. LLMs have excellent generating, summarizing, and programming abilities, making them competent for RIS function expansion assignments. Finally, when asked about the best approach for data analysis, 80% of subjects (16 out of 20) said the chat interface was more suitable (Table 3). This finding is also consistent with the chat interface based on LLMs supporting RIS function expansion by calling APIs.

Table 3. User preferences for the best approach in the data analysis module. The majority of subjects (16 out of 20) found the ChatUI interface more suitable for data analysis tasks, regardless of the order in which they tested the interfaces. This finding supports the hypothesis that a Chat interface based on LLMs can effectively support RIS function expansion through API calls.

Experiment Order	ChatUI	GUI	Total
ChatUI->GUI	7	3	10
GUI->ChatUI	9	1	10
Total	16	4	20

6. Discussion

This study explores the integration of LLMs into RIS to enhance human–computer interaction and improve the overall user experience. By conducting a series of experiments, we aimed to assess the feasibility, learnability, operational efficiency, and potential for function expansion of an LLM-based ChatUI compared to a traditional GUI. This study's statistical results might show inefficiency significance given our study sample size of 26 subjects, but overemphasizing statistical significance and the resulting neglect of possible key potentials are inappropriate [41]. To explore the large language model's potential in

RIS, the study's statistical significance results could illustrate totally. The results of Study 1 demonstrate that the ChatUI can effectively support core RIS operations, such as examination appointment, report retrieval, and data analysis. This finding aligns with recent research highlighting the potential of LLMs in various healthcare applications, including radiology report summarization [21] and medical question answering [37]. The successful completion of tasks using the ChatUI suggests that LLMs can be seamlessly integrated into RIS to provide a more intuitive and user-friendly interface. Study 2 investigated the learnability of the ChatUI compared to the GUI. The results indicate that users found the ChatUI easier to learn, with a lower learning difficulty and fewer instances of referring to guidelines. This finding is consistent with the notion that natural language interfaces can reduce the cognitive load associated with learning complex systems [10]. The ChatUI's ability to guide users through tasks using prompts and context-aware responses likely contributed to its enhanced learnability. In Study 3, we examined the operational efficiency of the ChatUI and GUI interfaces. The results showed that users encountered fewer errors and found the ChatUI more convenient to operate, despite being more proficient with the GUI. This finding suggests that the ChatUI's streamlined interaction style and context-aware guidance can lead to improved operational efficiency, even for users accustomed to traditional interfaces. However, it is important to note that network latency resulted in longer operation times for the ChatUI. Future implementations should consider optimizing network performance to fully realize the efficiency benefits of LLM-based interfaces. Study 4 explored the potential for expedient function expansion using the ChatUI. By integrating a data analysis module into the RIS demo program, we demonstrated that the ChatUI could leverage LLMs' generating, summarizing, and programming abilities to extend RIS functionality through API calls. Users found the ChatUI's presentation of results, which included both charts and text, more intuitive and suitable for data analysis tasks. This finding highlights the versatility of LLMs in supporting not only core RIS functions but also advanced analytical capabilities, aligning with recent research on LLMs' proficiency in reasoning and planning tasks [15,16].

The results of this study have several implications for the design and implementation of RIS interfaces. First, the integration of LLMs into RIS through a ChatUI can provide a more user-friendly and intuitive experience, potentially reducing training time and increasing adoption rates. Second, the ChatUI's ability to guide users through tasks and minimize errors can lead to improved operational efficiency and reduced cognitive load. Finally, the ChatUI's potential for expedient function expansion through API calls highlights the flexibility and scalability of LLM-based interfaces in accommodating evolving RIS requirements.

The straightforward approach of incorporating LLMs into Radiology Information Systems (RIS) via prompting engineering aims to explore the potential of enhancing interaction models within RIS systems, thereby unleashing prospects for increased efficiency and functionality. Nonetheless, the genuine endeavor of embedding LLMs into RIS as a novel paradigm for medical imaging systems encounters numerous significant challenges, including sufficient quantity and quality data, domain-specific knowledge deficiency and poor explainable, integration with existing systems, and ethical considerations.

- Data quality and quantity. Medical imaging datasets must encapsulate a broad spectrum of patient demographics and disease stages to ensure the LLM's performance is universally applicable and not skewed towards a particular subset. The complexity arises from overcoming biases inherent in existing datasets [42]. Biased data can lead to LLMs performing poorly for underrepresented patient groups, exacerbating healthcare inequalities. As LLMs learn from vast amounts of text data, integrating them with medical imaging necessitates meticulous calibration to interpret images accurately across diverse populations. This includes addressing issues like age, sex, ethnicity variations, and the progression of diseases, as highlighted in studies [43,44].
- Domain-specific knowledge and explainability. Medicine is an intensely specialized field that necessitates profound expertise and extensive clinical experience. AI-

though LLMs can process and comprehend vast amounts of textual information, their comprehension of the deeper intricacies of medical domains, such as pathophysiological mechanisms and rare disease manifestations, may be limited, leading to knowledge gaps in analyzing complex cases. Medical image diagnosis is not merely image recognition; it is firmly rooted in the clinical context and the experiential judgment of physicians. LLMs, due to the absence of realistic clinical encounters and accumulated experience, may struggle to replicate the intuitive judgments human doctors make based on experience, impacting the accuracy and credibility of diagnoses in specific scenarios. Furthermore, the explainability of diagnostic results is vital in clinical practice, influencing both physician trust in LLMs' decision-making and patient acceptance. Presently, the diagnostic reasoning produced by LLMs tends to be abstract, offering little insight into its logical pathway, which creates a considerable impediment in medical decision-making.

- **Integration with existing systems.** The primary obstacle to the integration of LLMs into existing medical imaging systems is the infrastructure demands. To fully leverage the potential of medical data, high-performance hardware is indispensable. This hardware must facilitate real-time processing of complex computations without disruptions, requiring substantial investments to establish a robust infrastructure against system failures [45]. Complementary to hardware, the development of software that is not only technically robust, but also seamlessly integrates into existing radiological workflows, is crucial. It is necessary to build close collaboration between software engineers, radiologists, and healthcare specialists to ensure software reliability and compatibility, thereby avoiding disruptions to established clinical practices [46].
- **Ethical considerations.** The integration of large language models into medical imaging elicits a myriad of ethical considerations, chiefly concerning data privacy, patient autonomy, the potential for misdiagnoses, and the balance between AI and human interaction in healthcare. Privacy and security issues are accentuated, especially when private entities handle sensitive medical data, risking unauthorized access or misuse [47]. Misdiagnosis risks introduce ethical dilemmas surrounding liability and accountability. Despite AI's diagnostic prowess, biases and errors can lead to patient harm, underscoring the urgency for clear guidelines outlining the roles and responsibilities of healthcare professionals and AI systems in diagnostic decision-making [48].

7. Conclusions

This study has undertaken an initial exploration into the integration of LLMs into RIS, focusing on the development of a conversational interactive interface using natural language. Through an exploratory study examining key RIS components such as examination appointment, report retrieval, and data analysis, we sought to evaluate the impact of this interface on learning and operational efficiency within the RIS context. Our findings suggest that by incorporating LLMs into three key components—understanding task requirements, planning task solutions, and leveraging APIs for coding and execution—significant advancements can be made in completing core functions of the RIS. The integration of LLMs into the RIS interface offers promising prospects for improving both the learning experience of trainees and the operational efficiency of practitioners. Specifically, our study reveals that the utilization of LLMs can simplify the learning process by providing intuitive interfaces that facilitate comprehension of complex RIS functionalities. Furthermore, the integration of LLMs into the RIS interface can streamline operational workflows, reducing the time required to perform tasks and minimizing the occurrence of errors. Moreover, the study highlights the potential for LLMs to extend the functional capabilities of RIS through the seamless integration of APIs. By leveraging LLMs' generation, summarization, and planning abilities, RIS systems can enhance their analytical capabilities and provide more actionable insights for practitioners. While this study provides valuable insights into the potential of LLM integration in RIS, it is important to acknowledge its limitations.

The sample size was relatively small, and the study focused on a specific set of RIS components. Future research should explore the long-term implications of LLM integration, examine its impact on a wider range of RIS functionalities, and investigate its scalability in real-world clinical settings. In conclusion, the findings of this study underscore the transformative potential of integrating LLMs into RIS interfaces, offering new avenues for improving medical training, diagnostic practices, and ultimately, patient care in the field of radiology.

Author Contributions: Conceptualization, Y.S. and L.Z.; methodology, L.Z. and Y.S.; software, J.H. (Jili Hu); validation, P.W. and J.H. (Junjun He); formal analysis, J.H. (Jili Hu); investigation, L.Z. and J.S.; resources, P.W.; data curation, L.Z. and F.L.; writing—original draft preparation, L.Z. and Y.S.; writing—review and editing, L.Z. and Y.S.; visualization, L.Z.; supervision, P.W.; project administration, J.S. and Y.S.; funding acquisition, P.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research was partly funded by the Central Government’s Special Fund for The Inheritance and Development of Traditional Chinese Medicine, grant number RZ2200001383.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are available from the corresponding author on reasonable request.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

HCI	Human–Computer Interaction
RIS	Radiological Information Systems
UI	User Interface
LLM	Large Language Model

References

- Gerlach, J.H.; Kuo, F.Y. Understanding human-computer interaction for information systems design. *MIS Q.* **1991**, *4*, 527–549. [[CrossRef](#)]
- Vallerio, K.S.; Zhong, L.; Jha, N.K. Energy-efficient graphical user interface design. *IEEE Trans. Mob. Comput.* **2006**, *5*, 846–859. [[CrossRef](#)]
- Ishwarya, M.; Anand, M.S.; Kumaresan, A.; Gopinath, N. Innovations in Artificial Intelligence and Human Computer Interaction in the Digital Era. In *Computational Imaging and Analytics in Biomedical Engineering*; Apple Academic Press: Palm Bay, FL, USA, 2024; pp. 105–145.
- Nance, J.W., Jr.; Meenan, C.; Nagy, P.G. The future of the radiology information system. *Am. J. Roentgenol.* **2013**, *200*, 1064–1070. [[CrossRef](#)]
- Rocchetti, M.; Delnevo, G.; Casini, L.; Mirri, S. An alternative approach to dimension reduction for pareto distributed data: A case study. *J. Big Data* **2021**, *8*, 39. [[CrossRef](#)]
- Lin, A.; Harris, M.; Zalis, M. Initial observations of electronic medical record usage during CT and MRI interpretation: Frequency of use and impact on workflow. *Am. J. Roentgenol.* **2010**, *195*, 188–193. [[CrossRef](#)]
- Nagy, P.G.; Warnock, M.J.; Daly, M.; Toland, C.; Meenan, C.D.; Mezrich, R.S. Informatics in radiology: Automated Web-based graphical dashboard for radiology operational business intelligence. *Radiographics* **2009**, *29*, 1897–1906. [[CrossRef](#)]
- Cowan, I.A.; MacDonald, S.L.; Floyd, R.A. Measuring and managing radiologist workload: Measuring radiologist reporting times using data from a Radiology Information System. *J. Med. Imaging Radiat. Oncol.* **2013**, *57*, 558–566. [[CrossRef](#)] [[PubMed](#)]
- Bender, E.M.; Gebru, T.; McMillan-Major, A.; Shmitchell, S. On the dangers of stochastic parrots: Can language models be too big? In Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency, Virtual, 3–10 March 2021; pp. 610–623.
- Orrù, G.; Piarulli, A.; Conversano, C.; Gemignani, A. Human-like problem-solving abilities in large language models using ChatGPT. *Front. Artif. Intell.* **2023**, *6*, 1199350. [[CrossRef](#)]
- Jiang, J.; Zhou, K.; Dong, Z.; Ye, K.; Zhao, W.X.; Wen, J.R. Structgpt: A general framework for large language model to reason over structured data. *arXiv* **2023**, arXiv:2305.09645.
- Sellam, T.; Das, D.; Parikh, A.P. BLEURT: Learning robust metrics for text generation. *arXiv* **2020**, arXiv:2004.04696.

13. Narayan, S.; Cohen, S.B.; Lapata, M. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. *arXiv* **2018**, arXiv:1808.08745.
14. Nallapati, R.; Zhou, B.; Gulcehre, C.; Xiang, B. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv* **2016**, arXiv:1602.06023.
15. Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Xia, F.; Chi, E.; Le, Q.V.; Zhou, D. Chain-of-thought prompting elicits reasoning in large language models. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 24824–24837.
16. Jiang, X.; Dong, Y.; Wang, L.; Shang, Q.; Li, G. Self-planning code generation with large language model. *arXiv* **2023**, arXiv:2303.06689.
17. Sanh, V.; Webson, A.; Raffel, C.; Bach, S.H.; Sutawika, L.; Alyafeai, Z.; Chaffin, A.; Stiegler, A.; Scao, T.L.; Raja, A.; et al. Multitask prompted training enables zero-shot task generalization. *arXiv* **2021**, arXiv:2110.08207.
18. Zhao, Y.; Khalman, M.; Joshi, R.; Narayan, S.; Saleh, M.; Liu, P.J. Calibrating sequence likelihood improves conditional language generation. In Proceedings of the Eleventh International Conference on Learning Representations, Kigali, Rwanda, 1–5 May 2023.
19. Nakano, R.; Hilton, J.; Balaji, S.; Wu, J.; Ouyang, L.; Kim, C.; Hesse, C.; Jain, S.; Kosaraju, V.; Saunders, W.; et al. Webgpt: Browser-assisted question-answering with human feedback. *arXiv* **2021**, arXiv:2112.09332.
20. Petridis, S.; Terry, M.; Cai, C.J. Promptinfuser: Bringing user interface mock-ups to life with large language models. In Proceedings of the Extended Abstracts of the 2023 CHI Conference on Human Factors in Computing Systems, Hamburg, Germany, 23–28 April 2023; pp. 1–6.
21. Jeblick, K.; Schachtner, B.; Dexl, J.; Mittermeier, A.; Stüber, A.T.; Topalis, J.; Weber, T.; Wesp, P.; Sabel, B.O.; Ricke, J.; et al. ChatGPT makes medicine easy to swallow: An exploratory case study on simplified radiology reports. *Eur. Radiol.* **2023**, 1–9. [[CrossRef](#)] [[PubMed](#)]
22. Nabovati, E.; Vakili-Arki, H.; Eslami, S.; Khajouei, R. Usability evaluation of laboratory and radiology information systems integrated into a hospital information system. *J. Med. Syst.* **2014**, *38*, 35. [[CrossRef](#)] [[PubMed](#)]
23. Dias, C.R.; Pereira, M.R.; Freire, A.P. Qualitative review of usability problems in health information systems for radiology. *J. Biomed. Inform.* **2017**, *76*, 19–33. [[CrossRef](#)]
24. Jansson, M.; Liisanantti, J.; Ala-Kokko, T.; Reponen, J. The negative impact of interface design, customizability, inefficiency, malfunctions, and information retrieval on user experience: A national usability survey of ICU clinical information systems in Finland. *Int. J. Med. Inform.* **2022**, *159*, 104680. [[CrossRef](#)]
25. Mann, K.S.; Bansal, A. HIS integration systems using modality worklist and DICOM. *Procedia Comput. Sci.* **2014**, *37*, 16–23. [[CrossRef](#)]
26. Mervak, B.M.; Davenport, M.S.; Flynt, K.A.; Kazerooni, E.A.; Weadock, W.J. What the patient wants: An analysis of radiology-related inquiries from a web-based patient portal. *J. Am. Coll. Radiol.* **2016**, *13*, 1311–1318. [[CrossRef](#)] [[PubMed](#)]
27. Calisto, F.M.; Nunes, N.; Nascimento, J.C. Modeling adoption of intelligent agents in medical imaging. *Int. J. Hum.-Comput. Stud.* **2022**, *168*, 102922. [[CrossRef](#)]
28. Mese, I.; Taslicay, C.A.; Sivrioglu, A.K. Improving radiology workflow using ChatGPT and artificial intelligence. *Clin. Imaging* **2023**, *103*, 109993. [[CrossRef](#)] [[PubMed](#)]
29. Russe, M.F.; Fink, A.; Ngo, H.; Tran, H.; Bamberg, F.; Reiser, M.; Rau, A. Performance of ChatGPT, human radiologists, and context-aware ChatGPT in identifying AO codes from radiology reports. *Sci. Rep.* **2023**, *13*, 14215. [[CrossRef](#)]
30. Hosny, A.; Parmar, C.; Quackenbush, J.; Schwartz, L.H.; Aerts, H.J. Artificial intelligence in radiology. *Nat. Rev. Cancer* **2018**, *18*, 500–510. [[CrossRef](#)]
31. Chang, Y.; Wang, X.; Wang, J.; Wu, Y.; Yang, L.; Zhu, K.; Chen, H.; Yi, X.; Wang, C.; Wang, Y.; et al. A survey on evaluation of large language models. *ACM Trans. Intell. Syst. Technol.* **2023**, *15*, 39. [[CrossRef](#)]
32. Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; Sutskever, I. Language models are unsupervised multitask learners. *OpenAI Blog* **2019**, *1*, 9.
33. Bahdanau, D.; Cho, K.; Bengio, Y. Neural machine translation by jointly learning to align and translate. *arXiv* **2014**, arXiv:1409.0473.
34. Lin, S.; Hilton, J.; Evans, O. Truthfulqa: Measuring how models mimic human falsehoods. *arXiv* **2021**, arXiv:2109.07958.
35. Nov, O.; Singh, N.; Mann, D. Putting ChatGPT's medical advice to the (Turing) test: Survey study. *JMIR Med. Educ.* **2023**, *9*, e46939. [[CrossRef](#)] [[PubMed](#)]
36. Yang, K.; Ji, S.; Zhang, T.; Xie, Q.; Ananiadou, S. On the evaluations of chatgpt and emotion-enhanced prompting for mental health analysis. *arXiv* **2023**, arXiv:2304.03347.
37. Singhal, K.; Tu, T.; Gottweis, J.; Sayres, R.; Wulczyn, E.; Hou, L.; Clark, K.; Pfohl, S.; Cole-Lewis, H.; Neal, D.; et al. Towards expert-level medical question answering with large language models. *arXiv* **2023**, arXiv:2305.09617.
38. Pellegrini, C.; Özsoy, E.; Busam, B.; Navab, N.; Keicher, M. RaDialog: A Large Vision-Language Model for Radiology Report Generation and Conversational Assistance. *arXiv* **2023**, arXiv:2311.18681.
39. Wang, X.; Wang, Z.; Liu, J.; Chen, Y.; Yuan, L.; Peng, H.; Ji, H. Mint: Evaluating llms in multi-turn interaction with tools and language feedback. *arXiv* **2023**, arXiv:2309.10691.
40. Liang, Y.; Wu, C.; Song, T.; Wu, W.; Xia, Y.; Liu, Y.; Ou, Y.; Lu, S.; Ji, L.; Mao, S.; et al. Taskmatrix. ai: Completing tasks by connecting foundation models with millions of apis. *arXiv* **2023**, arXiv:2303.16434.
41. Amrhein, V.; Greenland, S.; McShane, B.B. Scientists rise up against statistical significance. *Nature* **2019**, *567*, 305–307. [[CrossRef](#)]

42. Najjar, R. Redefining Radiology: A Review of Artificial Intelligence Integration in Medical Imaging. *Diagnostics* **2023**, *13*, 2760. [[CrossRef](#)] [[PubMed](#)]
43. Jayakumar, S.; Sounderajah, V.; Normahani, P.; Harling, L.; Markar, S.R.; Ashrafian, H.; Darzi, A. Quality assessment standards in artificial intelligence diagnostic accuracy systematic reviews: A meta-research study. *NPJ Digit. Med.* **2022**, *5*, 11. [[CrossRef](#)]
44. Roberts, M.; Driggs, D.; Thorpe, M.; Gilbey, J.; Yeung, M.; Ursprung, S.; Aviles-Rivero, A.I.; Etmann, C.; McCague, C.; Beer, L.; et al. Common pitfalls and recommendations for using machine learning to detect and prognosticate for COVID-19 using chest radiographs and CT scans. *Nat. Mach. Intell.* **2021**, *3*, 199–217. [[CrossRef](#)]
45. Barragán-Montero, A.; Javaid, U.; Valdés, G.; Nguyen, D.; Desbordes, P.; Macq, B.; Willems, S.; Vandewinckele, L.; Holmström, M.; Löfman, F.; et al. Artificial intelligence and machine learning for medical imaging: A technology review. *Phys. Medica* **2021**, *83*, 242–256. [[CrossRef](#)]
46. Wang, G. A perspective on deep imaging. *IEEE Access* **2016**, *4*, 8914–8924. [[CrossRef](#)]
47. Murdoch, B. Privacy and artificial intelligence: Challenges for protecting health information in a new era. *BMC Med. Ethics* **2021**, *22*, 122. [[CrossRef](#)]
48. Pinykh, O.S.; Lings, G.; Dewey, M.; Enzmann, D.R.; Herold, C.J.; Schoenberg, S.O.; Brink, J.A. Continuous learning AI in radiology: Implementation principles and early applications. *Radiology* **2020**, *297*, 6–14. [[CrossRef](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.