Review

Task Automation Intelligent Agents: A Review

Abdul Wali, Saipunidzam Mahamad * and Suziah Sulaiman *

Department of Computer & Information Sciences, Universiti Teknologi PETRONAS, Seri Iskandar 32610, Malaysia; abdul_21002750@utp.edu.my
* Correspondence: saipunidzam_mahamad@utp.edu.my (S.M.); suziah@utp.edu.my (S.S.)

Abstract: As technological advancements increase exponentially, mobile phones become smarter with machine learning and artificial intelligence algorithms. These advancements have allowed mobile phone users to perform most of their daily routine tasks on mobile phones; tasks performed in daily routines are called repetitive tasks and are performed manually by the users themselves. However, machine learning and artificial intelligence have enabled those tasks to be performed automatically, known as task automation. The users can perform task automation, e.g., through creating automation rules or an intelligent agent, e.g., conversational agents, virtual personal assistants, etc. Several techniques to achieve task automation have been proposed, but this review shows that task automation by programming by demonstration has had massive developmental growth because of its user-centered approach. Apple Siri, Google Assistant, MS Cortana, and Amazon Alexa are the most known task automation agents. However, these agents are not widely adopted because of their usability issues. In this study, two research questions are evaluated through the available literature to expand the research on intelligent task automation agents: (1) What is the state-of-the-art in task automation agents? (2) What are the existing methods and techniques for developing usability heuristics, specifically for intelligent agents? Research shows groundbreaking developments have been made in mobile phone task automation recently. However, it must still be conducted per usability principles to achieve maximum usability and user satisfaction. The second research question further justifies developing a set of domain-specific usability heuristics for mobile task automation intelligent agents.

Keywords: HCI; mobile; task automation; intelligent agent; usability heuristics; domain-specific heuristics; multimodal interaction

Citation: Wali, A.; Mahamad, S.; Sulaiman, S. Task Automation Intelligent Agents: A Review. *Future Internet* **2023**, *15*, 196. https://doi.org/10.3390/fi15060196

Academic Editor: Maria Gabriella Xibilia

Received: 3 April 2023
Revised: 17 May 2023
Accepted: 22 May 2023
Published: 29 May 2023

1. Introduction

Task automation automates tasks using technology that humans would otherwise perform. Task automation has the potential to improve efficiency, reduce errors, and free up time for more complex and creative work, therefore becoming significantly important [1]. Automating tasks can improve efficiency by reducing the time and effort required to complete them, allowing users to focus on more high-level and value-added tasks. Task automation also increases accuracy due to being less prone to errors and mistakes than manual tasks, which can improve quality and user satisfaction. Currently, task automation is being used in a wide range of industries, including manufacturing, healthcare, finance, and customer service. For example, in manufacturing, task automation leads to streamlining production processes, reducing errors, and increasing efficiency [2]. In healthcare, task automation helps improve patient care by automating the administering of medication and tracking patients' vital signs [3]. In finance, task automation processes transactions, generates reports, and performs other tasks efficiently [4]. In customer service, task automation is used to provide quick and accurate responses to customer inquiries through the use of chatbots [5], virtual personal assistants/intelligent agents [6], or recommender systems [7].

Over the last decade, the usage of conversational agents such as virtual personal assistants and intelligent agents has increased exponentially [8,9]. These conversational agents assist their users with task automation for personal and work-related activities, as mobile phones become the primary medium to conduct a variety of daily life activities and tasks, e.g., information requests (checking the weather, internet surfing, etc.); purchases (online shopping, ordering food, etc.); communication (instant text messages, calling, sharing on social media, etc.); carrying out professional work-related duties (health monitoring, remote server configuration, online teaching, etc.) [10]. Worldwide developments in machine learning and artificial intelligence and their integration with user interfaces have created a new field of studies which is known as intelligent user interfaces [11]. Advances in intelligent user interfaces have shown that these tasks could be automated and performed by virtual personal assistants or intelligent agents by giving natural language commands [12]. The most used intelligent agents are Apple Siri, Google Assistant, Microsoft Cortana, and Amazon Alexa, which can converse with users and provide daily updates and perform tasks on the user's behalf, e.g., checking the weather, sending a text message, searching the Internet, and conversing with children or adults [13].

With such diverse usage, the current intelligent agents also have their limitations; they directly call back-end services to activate underlying capabilities. Therefore, each supported application and service needs its own set of customized agents; they cannot control arbitrary third-party applications and services and can only, by default, activate built-in applications, e.g., messaging, calling, calendars, music, etc. At the same time, other intelligent agents can only access some integrated external applications and online services, e.g., searching Google, checking the weather, etc., as concluded by the study of [14]. To overcome the limitations of existing intelligent agents, [15,16] developed an intelligent agent that uses programming by demonstration, conversational approaches, learning from demonstration and natural language instructions, and generating an automation script from demonstrations on a graphical user interface. Programming by demonstration is an end-user development approach that allows users to teach computers and systems to perform certain tasks, enabling the creation of personalized systems or task automation [17].

Due to the end-user development approach, programming-by-demonstration systems have received greater attention in recent decades, and much work has been carried out regarding task automation using programming-by-demonstration. An earlier study review by [18] delivered a critique on programming-by-demonstration systems despite their potential to assist humans in daily life; their usage still needs to be improved, suggesting that it is primarily because of their poor usability. A recent systematic mapping study by Barricelli et al. [19] suggested that only a few unifying frameworks and approaches are available for guiding novice designers and practitioners in developing easy-to-use and easy-to-interact programming-by-demonstration task automation systems. Similarly, a study by Moussawi [20] concluded that continuous interaction with intelligent agents via voice helps users in various manners, i.e., it satisfies other non-utilitarian needs of users; cognitive affordance improves the usability of intelligent agents by boosting the user's efficiency and satisfaction, and functional affordance enhances the usefulness of intelligent agents. Therefore, several researchers and developers have worked on proposing task automation systems and intelligent agents using the programming-by-demonstration approach; however, little work has been carried out that focuses on enhancing the usability of these intelligent agents and intelligent systems.

Usability is a crucial factor to consider in designing task automation intelligent agents, as it can significantly affect the user experience and overall effectiveness of the systems [21,22], such that good usability can lead to higher user satisfaction, increased productivity, and improved efficiency [21]. Conversely, poor usability can lead to frustration, decreased productivity, and decreased efficiency [11]. Among available usability evaluation methods and techniques, heuristic evaluation and surveys/questionnaires are widely accepted methods within academia and industry [22]. Heuristic evaluation is an inspection approach that identifies usability problems based on usability heuristics or principles/guidelines. Heuristic evaluation entails usability experts inspecting a product's interface based on heuristics and identifying usability issues, which are then linked with usability heuristics. The experts evaluate each problem's frequency, severity, and criticality, which the developers correct. Heuristic evaluation is preferable due to its low cost compared to other methods in terms of time, usability experts, and resources; its minimal planning requirements; its applicability in the preliminary stages of software development, from paper prototype to executable systems; its ability to identify many problems, critical and less critical; and its avoidance of user involvement.

Currently, there are no specified usability heuristics for task-automating intelligent agents to perform heuristic evaluation because it is still in its preliminary stages, as concluded by the research of Elshan et al. [12]. Similarly, other methods such as usability test questionnaires, e.g., SUS (system usability scale), SUMI (software usability measurement inventory), and QUIS (questionnaire for user interaction satisfaction), cannot be used to evaluate task automation intelligent agents because they do not take into account the features of automation systems, their autonomy, or the intricate internal information processes, as concluded by Maehigashi et al. [23].

In summary, this study primarily investigates the following research questions:

- RQ.1. What is the state-of-the-art in task automation intelligent agents, and do these intelligent agents use any usability guidelines in the development process?
- RQ.2. What are the existing methods and techniques for developing usability heuristics, and for which domains have they been developed? Are there any domain-specific usability heuristics for evaluating intelligent agents?

Developing usability heuristics for task automation intelligent agents is important because of the nature of intelligent agents, which requires interaction with humans; this interaction should be natural and intuitive, and the agent should be able to understand and respond to user input in a way that is accurate and reliable, also ensuring that the agents are user-friendly and easy to use. Developing domain-specific usability heuristics for task automation intelligent agents can help ensure that these agents meet the requirements and provide guidelines for designing agents that are understandable and easy to use. Overall, developing usability heuristics for task automation intelligent agents is crucial and necessary to provide a positive and effective human–computer interaction experience.

2. Methodology

This study uses the search strategy previously used by Qiu et al. [24]. This is a systematic methodology that includes conducting a search on databases with search terms and analyzing articles based on inclusion and exclusion criteria. After the articles are analyzed, a full-text review is conducted on selected articles.

2.1. Data Collection

In April 2022, a literature search explored the available intelligent agents for task automation from 1994 to 2023. Similarly, a literature search explored the available usability heuristics from 2018 to 2023. The search for usability heuristics has been conducted since 2018 because a systematic literature review was already effectuated by Quiñones and Rusu [25].

2.1.1. Search Terms

The search terms used for the five databases were "Intelligent Task Automation Agents" and "Usability Heuristics OR Intelligent Agent". Both searched terms were modified according to the databases in different combinations.

2.1.2. Databases Searched

The relevant articles were searched on five different databases widely used by researchers of the human–computer interaction community: Scopus, Web of Science, Association for Computing Machinery (ACM) digital library, ScienceDirect, and the Institute of Electrical and Electronics Engineers (IEEE) Xplore. These research databases provide full-text journals and research papers published on intelligent agents, task automation, and usability heuristics.

2.2. Article Selection

This study follows the guidelines of the Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) statement, in an updated version by Page et al. [26]. The procedure of article selection is as follows:

A search strategy using the terms mentioned in Section 2.1.1 was conducted for articles from 1994–2023 for intelligent agents and usability heuristics from 2018–2023.

Duplicates were removed, and titles and abstracts were evaluated against inclusion and exclusion criteria.

Inclusion (quality criteria) and exclusion criteria are presented as follows:

- In inclusion or quality criteria, the articles were first selected based on the title. After the relevant titles were separated from the search results, the articles were filtered by reading the abstracts. Articles were included in the review if they had studies focusing on the system design of task automation intelligent agents, design of intelligent personal agents/assistants, intelligent virtual assistants, virtual personal assistants, user-centered design for task automation, or contained usability heuristics for intelligent agents, development of heuristics, proposed heuristics, user evaluations, or heuristics to evaluate intelligent user interfaces.
- In exclusion criteria, articles were excluded or rejected if they were not written in the English language, were duplicate reports of the same studies or systems from various sources, studies that had no design of intelligent agents for task automation, or studies that did not propose or develop any usability heuristics to evaluate intelligent agents or intelligent user interfaces.

2.3. Data Analysis

The articles for RQ.1 were coded in terms of (a) reference, (b) domain, (c) research aim, (d) findings, and (e) limitations. This review only analyzes the available task automation systems and intelligent agents using the programming-by-demonstration approach because of the user-centered design.

3. Results

3.1. RQ.1

To answer the research question (R.Q.1), twenty-one task automation systems and intelligent agents were selected for the review. Figure 1 visually represents the work undertaken in task automation using end-user development's programming-by-demonstration approach.

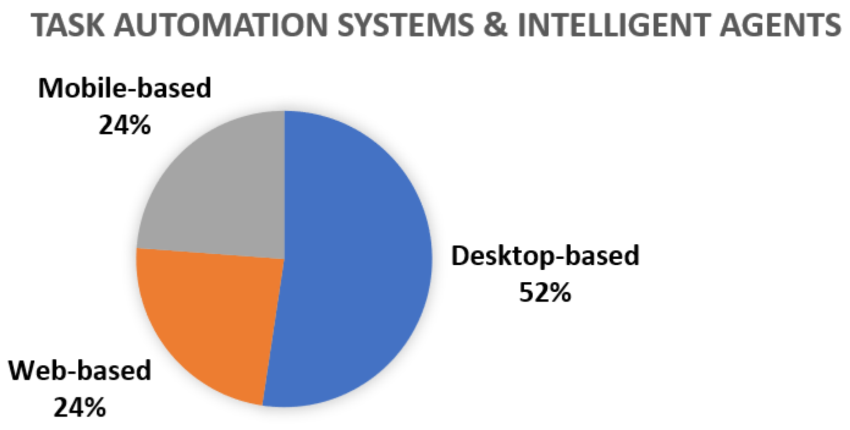

Figure 1. Task automation systems and intelligent agents.

3.1.1. Desktop-Based Task Automation Systems and Intelligent Agents

The early task automation systems and intelligent agents which were developed were mostly desktop-based. The first task automation system using the programming-by-demonstration approach was Pursuit by Modugno and Myers [27], which enables users to create abstract programs directly containing variables, loops, and conditionals within the interface. It made programming easy for the users but hardly automated their tasks and needed to be supported by a user study. Being the first of its kind, as a programming-by-demonstration system, Pursuit failed to adhere to standardized usability heuristics, and the user interface was developed based on the developers' experience. One of the main reasons is that Nielsen's heuristics were introduced the same year this system was developed, making it hard for researchers to consider usability [28]. SMARTedit, another programming-by-demonstration system, allowed users to automate repetitive text-editing tasks by learning techniques drawn from a machine-learning concept called version space algebra, through which it could learn useful text-editing procedures after only a few demonstrations. The limitations of this system were that it only worked for text editing, and no usability evaluation was performed [29]. Another task automation system that used a similar version of space algebra was CHINLE, which automatically constructed programming-by-demonstration systems for applications based on the interface specifications by Chen and Weld [30]. CHINLE was also unsupported by usability evaluation or user study, built atop SUPPLE, had fixed-length loops, and could not use logical connectives inside conditionals. DocWizard, another programming-by-demonstration system, presented a novel algorithm in the Eclipse platform for automatically capturing follow-me documentation wizards by demonstrations through observing experts performing procedures. The limitation of this system was that it lacked the features for the author to specify where user inputs were required and violated several usability principles, e.g., visibility of system status, user control and freedom, and error prevention [31]. FlashFill was inductive programming to create opportunities for task automation for non-programmers through synthesizing functional or logic programs for general-purpose tasks. However, the major challenges of this research were compositionality (i.e., requiring more usability studies) and making inductive programming more cognitive (i.e., ease of use or a simple and minimalistic design). Other challenges were domain change, validation, and noise tolerance [32].

Photo manipulation tutorials, where the author demonstrates the manipulation using an instrumented version of GIMP that records all the interface and application state changes. The system automatically generates tutorials from the recordings that illustrate the manipulation using images, text, and annotations [33]. These tutorials are not supported by usability evaluation and therefore have no feedback and error correction method; the semantic tool is based on computer vision and needs to learn macros from multiple demonstrations rather than one and to generalize it. Another GUI-based task automation system is Sikuli, which uses a similar visual approach to search and automate GUIs using screenshots. No usability evaluation was performed at any stage of development; therefore, it had the following issues and limitations: it did not work as expected due to theme variation and background changes, and it also had visibility constraints because it operated on visible screen elements and did not accept invisible GUI elements, e.g., hidden underneath other windows, tabs, or scrolling out of view [34]. Another GUI-based intelligent agent is Bespoke, a system synthesizing custom GUIs by observing user demonstrations of command-line applications. Theoretically and structurally, Bespoke seemed like a promising task automation system; however, in practice, a user study was performed instead of a usability study. A user study is equally important in the development of a system; however, it has some key changes, e.g., different goals or focus, methods, and stages in the design process. No testing of GUI synthesized by Bespoke on end-users to assess their benefits and limitations was performed; this is because the user study was conducted within a lab setting where the developers were present to guide study participants. Secondly, the authors needed a more rigorous assessment of user performance on a controlled set of tasks.

Lastly, no comparison of Bespoke against alternative GUI creation methods was conducted by Vaithilingam and Guo [35].

Ruler is an interactive visualization system synthesizing labeling rules using span-level interactive demonstration over document examples. It relieves users from the burden of writing labeling functions and enables them to focus on higher-level semantic analysis, such as identifying relevant signals for the labeling task. However, it is domain-dependent, as it is developed for data labeling, and therefore no usability evaluation was undertaken because of the specific domain dependency [36]. A state-of-the-art, desktop-based task automation system is Help-It-Looks-Confusing (HILC), a system prototype proposing a user-in-the-loop framework that learns to generate scripts of actions performed on the visual elements of GUI. A user study with the available baseline system Sikuli was conducted, which showed that Sikuli struggled to assist users in most of the test experiments. HILC accomplished simple linear and complicated tasks that spanned across multiple applications. The user study showed promising results favoring HILC; however, this state-of-the-art has limitations as well, e.g., basic actions are occasionally misclassified when none has a high probability; it works without the awareness of the state of the computer; it requires short fixed-length sleep commands after each action to account for computer loading time because the system cannot know if the operating system's task has finished or a webpage has loaded; the current appearance models have a fixed-size aspect ratio, which decreases the accuracy when items are of different sizes; processing a video tutorial takes a longer time because the system has to analyze every frame for the mouse and keyboard button status; propagation of errors to the pipeline due to inaccurate log-file generation, which is because the videos from the internet have noise and compression artifacts due to recording software and websites' video-sharing policies [37]. In the present study, the user study was centered solely on the end-user, with little attention paid to the system itself. The author contends that considering a usability study instead of a user study would have mitigated the identified limitations.

X-Droid is a framework that provides Android app developers with the ability to produce functional prototypes of applications quickly and easily. With this framework, developers can create a new app that imports various functionalities from other applications without understanding the implementation and source code. The limitation of X-Droid is responsiveness; it does not support reading images, sounds, or videos from the user interface; it does not support customized user interfaces such as interfaces managed by custom game engines; also, it can exploit external resources because the server cannot distinguish between X-Droid and regular users [38]. The researchers in this study performed a usability study to evaluate the application programming interface (API) and showed that the system was usable and easy to understand. However, as this is domain-specific and intended primarily for Android developers rather than the general population (i.e., normal users), it does not align with nor is it deemed a significant contribution to the research questions of this study. Nonetheless, it was noteworthy to mention the usability evaluation since the system was developed using a programming-by-demonstration (PbD) approach.

3.1.2. Web-Based Task Automation Systems and Intelligent Agents

d.mix is a tool for creating web mashups that leverages site-to-service correspondence. The user browses annotated websites and selects samples, and d.mix's sampling mechanism generates the underlying service calls that yield those elements. The limitations of this system are that the coexistence of two different sampling strategies confused the tool on how to separate a dataset; in a user study, participants had difficulty switching between multiple languages interspersed in a single page; documentation and error handling in the wiki environment was insufficient compared to other tools; and wiki-hosted applications were not scaled well beyond prototypes for a few users, similarly because a user study was performed instead of usability study which differs in goals, methods, and design process stages [39]. CoScripter developed a collaborative scripting environment for recording, automating, and sharing web-based processes [40]. A user study was performed instead of a usability evaluation for this task automation system. It was deployed in the office set-up, and over fifty corporate employees volunteered to incorporate it into their work practices. However, with usage over time, the issues of reliability and robustness provoked the need for advanced features due to upgrades in system interaction. Vegemite, an extension of CoScripter, was introduced with a spreadsheet-like environment that used direct manipulation and programming-by-demonstration techniques to populate tables with information collected from various websites automatically. However, the intelligent agent did not consider the response time of the web servers for requested data and did not support automatic or semi-automatic data cleaning [41].

Ringer is also a web-based task automation system in which a user demonstrates as input and creates a script that interacts with the page as a user would. The limitations of Ringer are that for action construct, some document object model (DOM) events occur at a remarkably high rate because JavaScript is single-threaded. A similar thread that records and replays each event must also process webpage interactions, so recording an exceptionally large number of high-frequency events can make pages slow to respond; the element construct's similarity-based node addressing approach is inherently best-effort but has no theoretical guarantees—however, in practice, it is sufficient; in the trigger construct, Ringer was designed for interactions that satisfy the trigger assumptions but fail when these do not hold. An overall limitation of Ringer is the possibility of failure due to client-side delays such as animations, timeouts, local storage issues, etc. [42]. Rousillon is a programming system based on relation selection and a generalization algorithm for writing complex web automation scripts by demonstration. This system allows the user to demonstrate how to collect the first row of a universal table view of the hierarchical dataset to teach Rousillon how to collect all rows. The limitation of Rousillon is that it focuses on realistic datasets, particularly distributed and hierarchical data [43].

3.1.3. Mobile-Based Task Automation Systems and Intelligent Agents

Assistive Macros by Rodrigues [44] presented an accessibility service to enable users to perform a sequence of commands with a single selection. The user could create these macros manually or automatically by detecting repeated interactions with a mobile phone. Assistive macros showed excellent results, but usability evaluation was not performed nor even a user study, and it was supported by only one case study; it also needed to support data with contextual information. Another mobile task automation system is InstructableCrowd, a crowdsourcing approach that allows users to create trigger-action (if, then) rules based on their needs via conversation [45]. The limitations of this system are that it does not focus on the robust creation of rules via conversation; it does not have a repository of common if-then patterns; it provides no feedback on the successful creation of rules; and there is no way to validate if-then rules created by the users. In addition, a user study was performed but not supported by a usability evaluation. VASTA, a vision and language-assisted programming-by-demonstration system for smartphone task automation overcomes three key challenges: (1) how to make a particular demonstration robust to positional and visual changes; (2) how to recognize changes in the automation parameters to make demonstration as generalizable as possible; (3) how to recognize from user utterance what automation the user wishes to carry out [46]. VASTA provides a vast domain usage due to the absence of complex engineering required to fetch, parse, and interpret various markup languages; vision-guided techniques can be used to supplement traditional methods; and it has no dependency on the user interfaces underlying markup languages. Therefore, it can be applied to systems where none is available, e.g., interfaces rendered with low-level graphic libraries. Similarly, as discussed above, this study is supported by a user study rather than a usability evaluation. The limitations of this intelligent agent are (1) semantic labeling of user interface elements: it only records a feature representation of user interface elements to track and find elements later while executing; (2) extensible markup language (XML) data with computer vision: automation might fail due to the object detection network's mistake. In the literature, two task automation intelligent agents are state-of-the-art in mobile phone task automation. One is Sugilite, and the other is DoThisHere.

Sugilite is a multimodal programming-by-demonstration system that enables users to create smartphone automation, which can be performed by giving voice commands. It uses the Android accessibility application programming interface (API) to support automating arbitrary tasks in any Android application. If the user gives a verbal command, which Sugilite does not know how to execute, it prompts the user to demonstrate by direct manipulating the regular apps on the user interface. Sugilite has some limitations, which are discussed in a dissertation by Li [14]. It has usability issues in text entry such as violating standard usability principles; another major concern is privacy and security while sharing generated scripts; it also does not support confirming crucial steps, e.g., online orders and purchases, and performing undo-able tasks which are also a violation of standard usability principles.

Figure 2 shows the working of Sugilite. The user interacts with the intelligent agent through voice commands. If the intelligent agent does not know how to perform a certain task, the user demonstrates the task on the user interface, and the intelligent agent observes and learns.

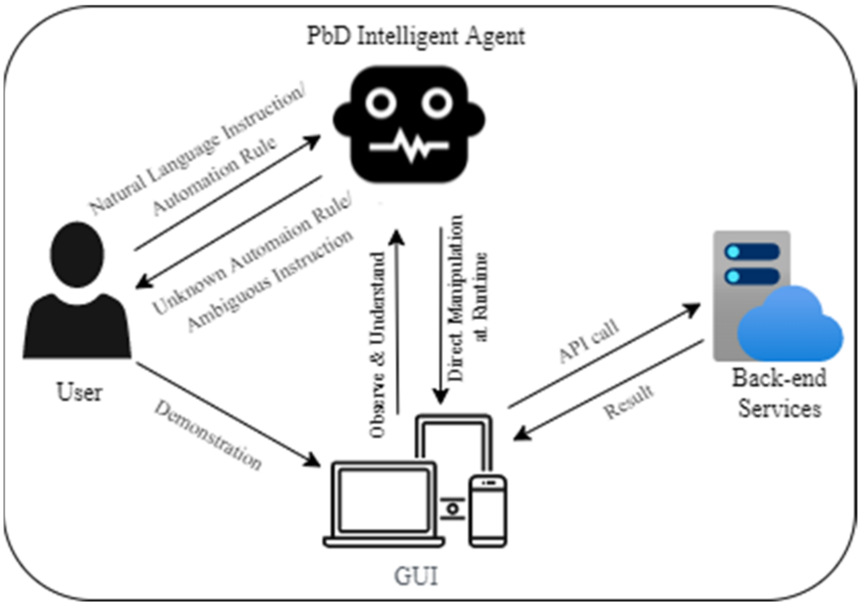
PbD Intelligent Agent
Natural Language Instruction/ Automation Rule
Unknown Automaion Rule/ Ambiguous Instruction
User
Demonstration
Observe & Understand
Direct Manipulation at Runtime
API call
Result
Back-end Services
GUI

Figure 2. Sugilite learning task automation with a programming-by-demonstration (PbD) approach.

Similarly, DoThisHere is also a multimodal interaction technique that streamlines cross-app tasks and reduces the burden of performing tasks imposed on users. It allows users to use voice commands to refer to information or app features that are off-screen and touch to specify where the relevant information should be inserted or displayed. This allows users to transfer information to other apps with less context switching [47]. The user study showed that several of the system's features were not used in the user study. Secondly, time delays affect the user taps on the icon before the system can listen to voice commands and user interface selections. The limitation of this system is that it uses a virtual assistant framework design to handle ambiguity and provide feedback; it also requires improvement in the user interface element selection algorithm.

3.2. RQ.2

To answer the research question (*RQ.2.*), thirty-nine review papers were selected for full-paper review.

3.2.1. Heuristic Development Methodologies

Table 1 describes the methods and techniques for developing usability heuristics in the last half-decade.

Table 1. Heuristic development methods and techniques.

Heuristic Development Methodology		References	Number of Studies
Existing Heuristics	Nielsen's	[48–54]	7
	Others	[55–60]	6
Literature Review		[61–69]	9
Mixed Processes		[70–78]	9
Usability Problems		[79–85]	7
Theory		[86]	1
	Total		39

Similarly, Figure 2 visually represents the heuristic development methodology within the last half-decade.

Figure 2 describes the methods and techniques adopted by the available researchers to achieve their goals. In existing heuristics studies, researchers have used existing heuristics established by practitioners and designers to develop or propose their own domain-specific heuristics. In literature reviews, researchers have used the literature to develop heuristics. Similarly, researchers in mixed processes have used different methods and techniques to develop heuristics, i.e., heuristic evaluation, usability testing, questionnaires, interviews, experiments, or even tools. In usability problems, researchers conducted a user study, interviews, usability testing of existing products, etc., to develop heuristics. However, only one researcher developed usability heuristics using universal design theory.

3.2.2. Development of Heuristics in Domains

Among the heuristics developed, Table 2 shows the domains in which heuristics are developed or proposed.

Table 2. Domains of developed usability heuristics.

Domain	References	Number of Studies
Health Information Systems	[49,71]	2
Online Websites	[48,63,70,80]	4
Virtual Learning Environments	[51,53,64,69]	4
Mobile Applications	[50,54,56,58,59,62,65,68,76,79,82–84,87]	14
Intelligent Agents	[57,78,85]	3
Other Domains	[52,60,61,66–68,72–75,77,81]	12
Total		39

Figure 3 visualizes the work carried out in different domains to develop usability heuristics.

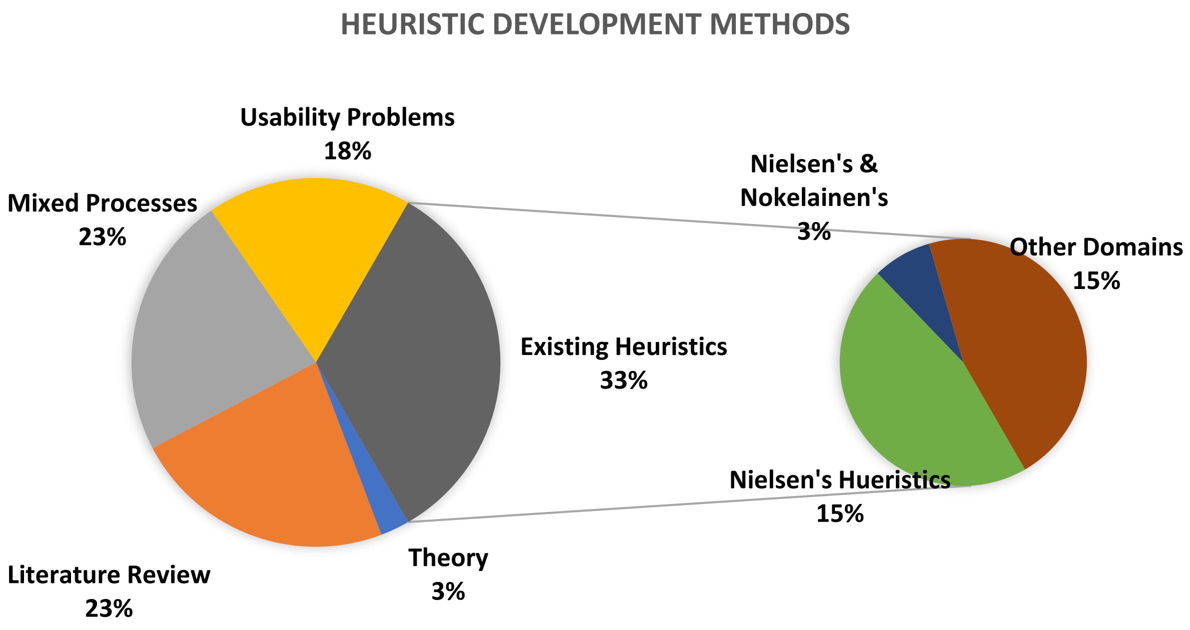

Figure 3. Heuristic development methods.

Figure 4 visualises the domains in which the usability heuristics have been developed in the last half decade.

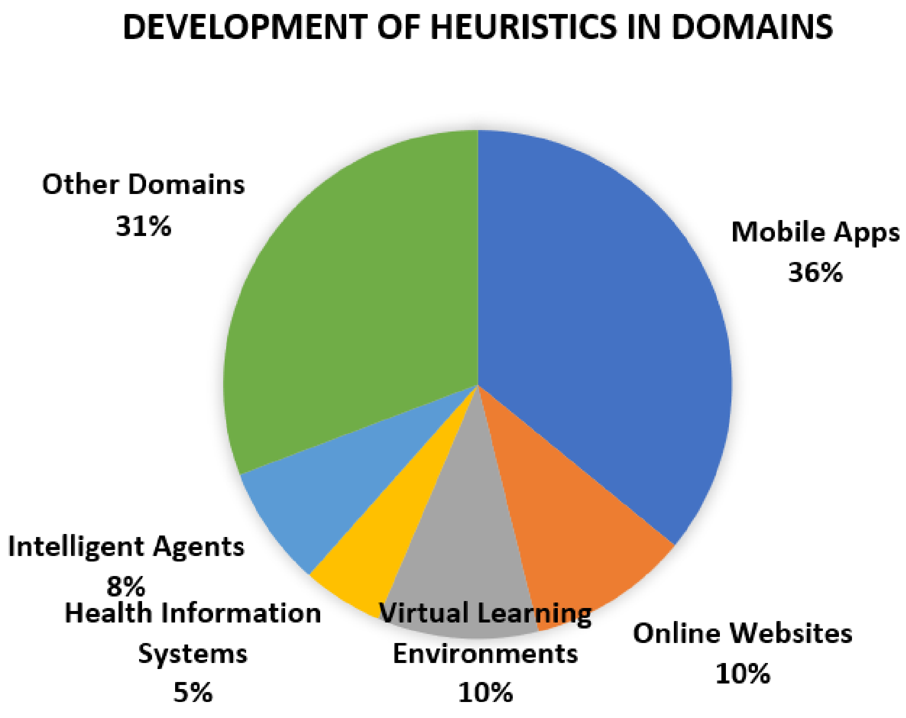

Figure 4. Development of heuristics in domains.

3.2.3. Analysis

Health Information Systems

Among the reviewed studies, only two research papers present work on the usability evaluation of health information systems. Tremoulet et al. [71] used a mixed method to develop usability heuristics for health information systems. A literature review was conducted, clinical interviews were taken to gather qualitative data, and a survey was conducted to gather quantitative data. The data gathered were reviewed, issues were identified within the health information system, and usability heuristics were generated based on the identified issues. The developed heuristics were then cross-checked with existing heuristics to eliminate the overlapping heuristics. The new heuristics were then implemented in the health information system, and validation was performed through heuristic evaluation by expert reviewers.

Another study by Bouraghi et al. [49] used Nielsen's existing heuristics to evaluate the health information system. They initially performed an expert evaluation of the system and identified usability errors, which were then categorized in severity. Validation was not performed because the study did not propose any heuristics.

Health information systems share several commonalities with intelligent agents, such as data processing, decision support, personalization, communication and interaction, integration with other systems, security, and privacy. While there are commonalities between intelligent agents and health information systems, it is important to note that they also have distinct features and purposes. Intelligent agents focus on intelligent decision-making and user interaction, while health information systems are specifically designed to manage and exchange healthcare-related data and support clinical workflows. Therefore, usability heuristics development methods for both domains could be similar, or general usability principles may overlap, but specific considerations and priorities reflect the unique characteristics and requirements of their respective domains and interaction modalities.

Online Websites

A systematic methodology for the development of usability heuristics by Quiñones and Rusu [25] has been adopted by several researchers to develop domain-specific usability heuristics, and Saavedra et al. [63] developed usability heuristics for social networks using a similar methodology. A domain-specific literature review was conducted, and the eight stages for systematic development of heuristics were performed. The developed heuristics were evaluated by experts, the results were analyzed, and future work was presented. However, Huang [80] performed a case study in which the author used mixed processes to evaluate tourism websites. A user-centered approach to empirically assess the website was undertaken, which involved conducting a literature review, a selection of websites, a selection of tasks to be performed by participants, development of a usability questionnaire for assessment, and data collection and analysis, and a discussion with results was conducted. In this work, no validation process was performed because the heuristics were generated after the literature review and experiment.

Similarly, Krawiec and Dudycz [48] used Nielsen's heuristics to evaluate a website and identify usability errors. After identification, errors were categorized, and new categories were added to Nielsen's heuristics. Experts evaluated the proposed heuristics, and results showed that new categories were able helpful in detecting more usability errors. A mixed development method was adopted by Zardari et al. [70] to develop usability heuristics for an e-learning website. The study included heuristic evaluation, usability testing, a user experience questionnaire, and eye tracking to identify usability errors. Heuristics were developed and used to identify errors. The designers and developers fixed the identified errors, and after fixing the errors, another usability test was completed to validate the proposed heuristics. A questionnaire survey was conducted to gather qualitative feedback.

Online websites and intelligent agents also share some commonalities, such as communication and interaction with diverse users, information access, personalization, automation, and integration. While there are similarities, it is also important to note the distinct characteristics and purposes. Online websites primarily serve as platforms for presenting information and conducting online transactions, while intelligent agents focus on conversational interactions, task automation, and providing personalized assistance. Therefore, usability heuristics development methods also require a focus on different priorities and considerations due to key differences in interaction modalities, information presentation, and user inputs.

Virtual Learning Environments

Usability heuristics to evaluate virtual labs were proposed by Kumar et al. [53] after conducting a literature review. These heuristics were a combination of Nielsen's and Nokelainen's heuristics. The researchers performed a literature review and selected heuristics relevant to virtual labs. The proposed heuristics were then evaluated using available virtual lab platforms, and an analysis was made. The results showed that proposed usability heuristics were more helpful in detecting usability issues. The proposed heuristics were not validated because they were developed after a literature review. A similar approach was adopted by Vieira et al. [64] to develop usability heuristics for evaluating the usability of educational games. The development method used in this research was based on a literature review. It was conducted in four phases: identification of articles to be selected for review, triage–study selection and exclusion, articles included after eligibility, and an analysis was conducted. The results showed that proposed heuristics could not be validated, as the field of knowledge was still beginning to develop.

Using Nielsen's heuristics, systematic usability heuristics were developed by Figueroa et al. [51]. This study also adopted the development method proposed by Quiñones and Rusu [25] and validated the heuristics through experiments, heuristic evaluation, case studies, and user tests. A post-pandemic study was conducted by Ismail et al. [69] to evaluate online learning environments such as Zoom and Teams. This study was initiated by conducting a literature review and selecting usability heuristics relevant to online learning environments. The proposed heuristics were used to evaluate the two most widely used platforms, and an analysis was conducted. No validation process of heuristics was carried out because heuristics were developed from the literature review.

The virtual learning environment and intelligent agents have similarities such as personalized learning, adaptive learning, providing in-time support and assistance, automation, and feedback and assessment. Despite similarities, virtual learning environments focus on providing a digital platform for instructional content delivery, interaction, and assessment. Intelligent agents, on the other hand, emphasize personalized assistance, automation, and conversational interactions for enhanced learning experiences. General usability principles may overlap intelligent agents, but specific considerations and priorities reflect each platform's unique characteristics and objectives.

Mobile Applications

In the last half-decade, massive mobile application growth has been observed across different fields and domains. Similarly, the human–computer interaction field has been trying to catch up with the fast-paced development in the mobile industry. Several usability heuristics and evaluation methods have been proposed and implemented by academicians and industry experts to enhance the usage of mobile phones, according to Hasan et al. [88].

A literature review conducted by Da Costa et al. [62] for the quality assessment of mobile phones proposed usability heuristics. These heuristics could not be validated, and the author intended to use them in empirical validation, allowing dynamic incorporation and improvements. Similarly, a literature review by Salah et al. [68] proposed usability heuristics for evaluating m-commerce applications in Arabic. This study also did not validate the proposed usability heuristics because the heuristics were developed after a literature review, and the authors planned to assess different interfaces in Arabic. Another study by Kumar et al. [76] developed usability heuristics for mobile learning applications using a mixed-process approach. They used Nielsen's heuristics and a literature review to develop specific heuristics, which were then used to categorize usability problems. An expert evaluation was performed to validate the proposed heuristics.

The only study that uses existing heuristics, except Nielsen's, was conducted by Sancho Nascimento et al. [59] to develop usability heuristics for mobile game applications for children with Down syndrome. This study used existing heuristics proposed by Breyer evaluation, Able Games Association, and Recommendations of Preece, Sharp, and Rogers to develop the game. The game was evaluated by usability experts and endorsed by a walkthrough with health professionals. Another usability heuristic was developed for mobile games by Robson and Sabahat [84]. They identified usability problems in existing game applications and analyzed the gathered data. After data analysis, usability heuristics were developed and implemented by creating a game prototype. The implemented usability heuristics were validated by expert gamers evaluating the game prototype.

A literature review was conducted by Abreu et al. [65] to evaluate the usability of children's education applications. Usability heuristics were developed by reviewing the literature, and expert evaluators and an experiment evaluated the developed heuristics. Expert evaluators included teachers, researchers in child education, specialists in HCI, and researchers in computing in education. Another study by Limtrairut et al. [56] proposed heuristics for an m-learning application and developed a prototype using existing heuristics. The proposed heuristics were validated by seven experts evaluating the prototype.

After conducting a literature review and questionnaire survey from users, design recommendations for mobile stock exchange applications were presented by Hussain et al. [58]. Expert evaluators evaluated the developed heuristics by reviewing the existing applications and identifying problems. The analysis was conducted, and design recommendations were presented as results. Similarly, usability heuristics to evaluate financial technologies were developed by Ali et al. [54] based on Nielsen's heuristics. Bashir et al. [55] presented usability heuristics for fitness-related, context-aware mobile applications based on existing usability heuristics. The researchers reviewed the literature, developed domain-specific heuristics, and evaluated existing applications. After the identification of the problem, the heuristics were refined and validated. The validation was performed through two evaluation studies and one usability expert review. A similar approach using Nielsen's heuristics was conducted by Faria Gomes et al. [50], but this study focused on IOS applications. The study was fairly similar in terms of methodology. However, the validation was performed by conducting a system usability scale (SUS) questionnaire.

For evaluating children's education relating to mobile applications, Samarakoon et al. [79] experimented with preschool children and observed while children interacted with the tablet's interface. The problems were identified, and new heuristics were developed based on the observations. The developed heuristics were implemented in the interface, and another experiment was conducted for validation. Similarly, Eltalhi et al. [82] evaluated children's education application in three steps: pre-test, post-test, and usability test. However, no validation was performed because this study did not propose usability heuristics.

One study validated usability heuristics during the design phase through user testing, including a mix of surveys, concurrent think-aloud, and interviews for feedback on the prototype. This study was conducted by Kim et al. [83] and evaluated a disease app. The methodology used to generate usability heuristics was based on usability problems gathered by creating personas, conducting competitor analysis, heuristic evaluation, and user interviews. Another study used universal design theory to propose usability heuristics. After conducting a literature review, heuristics were developed by the researcher. The developed heuristics were evaluated and validated by mixed methods, including usability testing and user-experience evaluation, and the designers addressed the results.

Usability heuristics for mobile applications and intelligent agents also share commonalities such as user-centered design, interaction design, error prevention and recovery, and consistency and familiarity. However, there are also some differences in terms of interaction modalities, context and portability, presentation and content, and multimodal capabilities. Similarly, both differ in unique characteristics and specific considerations.

Intelligent Agents

Usability heuristics for speech-based smart devices and intelligent agents differ based on their specific characteristics and intended uses. To evaluate the speech-based smart devices, a literature review was conducted on existing heuristics, and domain-specific heuristics were developed by Wei and Landay [57]. After the development of heuristics, an evaluation of speech-based smart devices was carried out. Expert evaluators validated proposed heuristics. Speech-based smart devices differ in terms of physical interaction. Usability heuristics for speech-based smart devices consider physical aspects of interaction, such as wake word detection, microphone sensitivity, and voice recognition accuracy.

Similarly, usability heuristics for voice user interfaces also differ from intelligent agents in terms of task-oriented interactions. To evaluate Voice User Interfaces (VUIs), a usability study was conducted using System Usability Scale (SUS), Post-Study System Usability Questionnaire (PSSUQ), heuristic questionnaires, and interviews by Pyae [85]. The data gathered from the usability study was analyzed, and usability heuristics were formulated from the analysis; therefore, no validation of heuristics was required. The usability heuristics for voice user interface focus on designing interactions for specific tasks or use cases, such as clear and concise prompts, appropriate dialog flow, and accurate recognition of user commands.

Usability heuristics to evaluate chatbots developed by Sánchez-Adame et al. [78]. In their study, the researchers conducted a literature review and developed usability heuristics based on their experience developing related applications and systems The researchers adopted a modified version of the Quiñones and Rusu [27], findings for this study and performed 6 stages of systematic heuristic development. Chatbots are also considered intelligent user interfaces, however, they differ from intelligent agents in terms of conversational flow. Usability heuristics for chatbots prioritize natural and coherent conversations. Addressing factors such as contextual understanding, maintaining conversational context, and generating appropriate responses based on user inputs.

While there may be some overlaps in general usability principles, the specific considerations and priorities in usability heuristics for speech-based smart devices, voice user interfaces, chatbots, and intelligent agents reflect the unique characteristics and objectives of each system or platform. Speech-based smart devices focus on physical and audio-related aspects, voice user interfaces emphasize task-oriented interactions, chatbots prioritize conversational flow, and intelligent agents aim for adaptive and personalized experiences.

Other Domains

Usability heuristics to evaluate information architecture framework for academic library websites are proposed by Silvis et al. [67]. The methodology used for heuristic development is based on a literature review. An analysis was conducted, proposed heuristics were implemented in websites, and recommendations and reviews were provided, but no heuristics were validated.

A tool to evaluate the usability of games using Nielsen's heuristics was used by Yanez-Gomez et al. [74], and usability problems were identified. A domain-specific heuristics were developed based on problems identified and implemented into the games. After developing heuristics were implemented, a preliminary evaluation was conducted on two games. Analysis suggested that the proposed heuristics successfully identified usability problems in the games.

Usability heuristics to evaluate the interface for Arabic m-commerce applications were developed by Salah et al. [68], which were also used to evaluate the system interface in the Arabic language by Muhanna et al. [60]. These heuristics were developed by conducting a systematic literature review, and usability experts validated the proposed heuristics. The developed heuristics successfully detected usability issues and violations in interfaces of the Arabic language.

A literature review on Nielsen's heuristics and user tests was conducted to develop usability heuristics for evaluating systems with tabletop interfaces by de Franceschi et al. [75]. This study adopted a modified version of Quiñones and Rusu [25], and a prototype was developed with proposed heuristics. The proposed heuristics were validated by a case study where multiple users used the prototype.

A set of usability heuristics developed by Umar et al. [52] focused on enhancing the usability of systems used for children's education, also known as child computer interaction

(CCI), using Nielsen's heuristics. This study was carried out by conducting a use-case study with children and identifying the usability issues. Nielsen's heuristics were modified and evaluated by experts. The finalized heuristics were implemented in a prototype and another round of user testing validated the heuristics. The prototype developed with the proposed heuristics was more usable for children's education.

For evaluating interactive web maps, Marquez et al. [66] used the eight stages proposed by Quiñones and Rusu [25] to develop usability heuristics systematically. The eight stages include exploratory study, experimentation, descriptive, correlation, selection, specification, validation, and refinement. The developed heuristics were validated by experts performing the heuristic evaluation.

To evaluate set-top box and television interfaces, Kaya et al. [73] performed a mixed process involving problems identified by the developers, three experts with cognitive walkthroughs, and customer complaints to develop usability heuristics. Based on the gathered data, the researchers developed usability heuristics, and experts evaluated those heuristics by creating clusters of problems previously identified. A validation checklist was created based on cluster analysis. The proposed heuristics were validated by user testing, expert judgment, and heuristic evaluation.

Viana et al. [81] applied usability heuristics in a machine-learning system for data labeling. A preliminary study was conducted to identify the usability problems in an existing system. The gathered data were analyzed and compared to Nielsen's heuristics. The finalized heuristics were used to develop the labeling system.

Existing heuristics could not be used for evaluating augmented reality (AR) or mixed reality (MR); therefore, Derby et al. [61] conducted a literature review to develop usability heuristics for these systems. For the heuristic development, the eight stages proposed by Quiñones and Rusu [25] were used, and expert reviews, heuristic evaluation, and user testing of the systems performed the validation. A similar approach of eight-stage heuristic development for evaluating progressive web applications was also adopted by Anuar et al. [72]; however, the validation was performed by five experts from academia and industry on three different domain applications: cultural heritage, stock photo industry, and marketplace.

Usability heuristics for evaluating a hospital-based computerized decision support system (CDSS) developed by Marcilly et al. [77] using a mixed approach, where the heuristic evaluation of the existing system was performed. Concurrently, the researchers conducted questionnaires and interviews with hospital staff. Cross-checking of collected data was performed, and after a comprehensive analysis, the heuristics were validated during the design phase through user testing.

4. Discussion

The field of intelligent task automation systems and intelligent agents has rapidly grown over the past few years. While most of these systems are currently desktop-based, recent advancements in mobile phone devices have led to the development of mobile-based task automation systems. However, despite these advancements, limitations still need to be addressed.

This review study concludes that one of the major limitations or causes of the inadaptability of such mobile-based task automation systems and applications is the unavailability of domain-specific usability heuristics for developers and designers to develop easy-to-use and user-friendly systems. Even with high-speed processors and RAMs, these systems can still present usability issues. Only if a user understands how a certain function works or why certain functions exist will they be able to use the device to its full potential. Otherwise, many of the system functionalities will not be known to the users due to the bad design of the interface. Additionally, even if users understand how to perform certain tasks and activities, they can only sometimes be sure that the system or application will perform as expected.

This study also suggests that the human–computer interaction community needs to give more attention to developing systematic domain-specific usability heuristics, such as for task automation intelligent agents, because these systems have the potential to make human life easier. To effectively utilize this potential, usability is an essential aspect to consider during the design and development of such systems and applications. In addition to usability issues, there are other limitations to mobile-based task automation systems and applications; for example, they may need help to handle large amounts of data or complex tasks as effectively as desktop-based systems. Additionally, they may need more battery life and storage capacity, which can limit their usefulness for certain tasks; this can be considered future work.

Despite these limitations, there is a significant amount of development in the field of mobile-based task automation systems and applications. However, as discussed in research question R.Q.2, most proposed or developed usability heuristics have been focused on domains other than task automation intelligent agents. This raises the question of why there is a significant amount of development in one area, while the human–computer interaction community is exploring other fields. While it is valuable to explore other fields, it is essential to consider the need for rapid advancements and the demands of daily life. Mobile technology has become an integral part of people's lives, and mobile-based task automation systems or applications have the potential to provide numerous benefits to users. Therefore, R.Q.2 concludes that it is essential to undertake systematic efforts to support the development of mobile technology and mobile-based task automation systems and applications.

The author also suggests some approaches to overcome the limitations and concludes from this review study that one potential approach to address these limitations is to design mobile-based task automation systems and applications with a user-centered approach. This approach involves involving users in the design process to ensure that their needs and preferences are taken into account. Additionally, usability testing can be conducted to identify potential issues and make necessary improvements before releasing the product. Another approach is to incorporate more machine learning and other artificial intelligence techniques into mobile-based task automation systems. These techniques can help improve the efficiency and effectiveness of these systems and applications, making them more useful for a wider range of daily life tasks. While there are limitations to mobile-based task automation systems, they have the potential to provide numerous benefits to users. To realize this potential, the human–computer interaction community needs to give attention to usability issues and undertake systematic efforts to support the development of mobile technology. By doing so, we can create mobile-based task automation systems that are efficient, effective, and user-friendly.

The development of usability heuristicsfor intelligent agents that automate tasks is essential due to the unique nature of these agents, which necessitates human interaction. This interaction should feel natural and intuitive, with the agent being capable of accurately and reliably understanding and responding to user inputs. It is also important to ensure that the agents are user-friendly and easily accessible. By creating domain-specific usability heuristics for task automation intelligent agents, we can guarantee that these agents meet the requirements and offer guidelines for designing agents that are comprehensible and straightforward to use. Ultimately, the development of usability heuristics for task automation intelligent agents is vital and indispensable for providing a positive and efficient human–computer interaction experience.

5. Conclusions

In conclusion, this study shows the availability and potential of a wide range of research work that could be carried out in this domain. This study also confirms the need for usability heuristics to be developed in the future to develop usable task automation intelligent agents effectively and efficiently. Developing usability heuristics for task automation intelligent agents is a vital aspect to consider while creating effective human–computer interaction experiences. These intelligent agents are designed to interact with humans, and thus, it is crucial that the interaction must be natural and intuitive. To achieve this, it is essential to develop usability heuristics that can guide the design process of these agents, ensuring that they are user-friendly, easy to use, and accurately respond to user inputs. These agents can be designed for a specific domain, such as healthcare, finance, or customer services, where the requirements for user interaction might vary, and domain-specific usability heuristics can be developed to ensure that the agents meet the necessary criteria.

The study also shows the development of usability heuristics for task automation intelligent agents and systems which considers the intelligence and automation aspects of the devices interacting with the users multimodally, e.g., voice, gestures, contextually aware techniques, etc. The usability heuristics for task automation intelligent agents should aim to provide guidelines for creating agents that are easy to learn and use, with a minimal cognitive load on the user. The heuristics should focus on aspects such as the visibility of system status, which ensures that the user is aware of the agent's current state and that the feedback provided to the user is relevant and timely. Additionally, the heuristics should aim to reduce the need for the user to remember complex commands or procedures, and instead, provide recognition-based interactions that allow users to easily recognize the desired action.

Overall, developing usability heuristics for task automation intelligent agents is crucial to ensure that these agents meet the user's needs, are easy to use, and provide a positive user experience. The heuristics can guide the design process and help create agents that accurately respond to user inputs while reducing cognitive load and ensuring user satisfaction.

Author Contributions: Conceptualization, A.W.; methodology, A.W.; validation; A.W., S.M. and S.S.; formal analysis, A.W.; investigation, A.W.; writing—original draft preparation, A.W.; writing—review and editing, A.W., S.M. and S.S.; supervision, S.M. and S.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research/paper was partially supported by Centre for Graduate Studies (CGS), Universiti Teknologi PETRONAS with cost centre (015BD1-001) and partially supported by Institute of Health and Analytics (IHA), Universiti Teknologi PETRONAS with Cost Centre (015NC0-001).

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare that they have no conflict of interest/competing interests.

References

1. Mohamed, S.A.; Mahmoud, M.A.; Mahdi, M.N.; Mostafa, S.A. Improving efficiency and effectiveness of robotic process automation in human resource management. *Sustainability* **2022**, *14*, 3920. [CrossRef]
2. Iqbal, J.; Islam, R.U.; Abbas, S.Z.; Khan, A.A.; Ajwad, S.A. Automating industrial tasks through mechatronic systems-a review of robotics in industrial perspective. *Teh. Vjesn. Tech. Gaz.* **2016**, *23*, 917–924.
3. Bauer, M.; Monteith, S.; Geddes, J.; Gitlin, M.J.; Grof, P.; Whybrow, P.C.; Glenn, T. Automation to optimise physician treatment of individual patients: Examples in psychiatry. *Lancet Psychiatry* **2019**, *6*, 338–349. [CrossRef] [PubMed]
4. Mehrotra, A. Artificial intelligence in financial services–need to blend automation with human touch. In Proceedings of the 2019 International Conference on Automation, Computational and Technology Management (ICACTM), London, UK, 24–26 April 2019; IEEE: Piscataway, NJ, USA, 2019. [CrossRef]
5. Stoilova, E. AI chatbots as a customer service and support tool. *ROBONOMICS J. Autom. Econ.* **2021**, *2*, 21.
6. Kepuska, V.; Bohouta, G. Next-generation of virtual personal assistants (microsoft cortana, apple siri, amazon alexa and google home). In Proceedings of the 2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC), Las Vegas, NV, USA, 8–10 January2018; IEEE: Piscataway, NJ, USA, 2018. [CrossRef]
7. Nunes, I.; Jannach, D. A systematic review and taxonomy of explanations in decision support and recommender systems. *User Model. User-Adapt. Interact.* **2017**, *27*, 393–444. [CrossRef]
8. Feine, J.; Gnewuch, U.; Morana, S.; Maedche, A. A taxonomy of social cues for conversational agents. *Int. J. Hum.-Comput. Stud.* **2019**, *132*, 138–161. [CrossRef]
9. Pfeuffer, N.; Benlian, A.; Gimpel, H.; Hinz, O. Anthropomorphic information systems. *Bus. Inf. Syst. Eng.* **2019**, *61*, 523–533. [CrossRef]

10. Deng, T.; Kanthawala, S.; Meng, J.; Peng, W.; Kononova, A.; Hao, Q.; Zhang, Q.; David, P. Measuring smartphone usage and task switching with log tracking and self-reports. *Mob. Media Commun.* **2019**, *7*, 3–23. [CrossRef]
11. Schmidt, A.; Mayer, S.; Buschek, D. Introduction to Intelligent User Interfaces. In Proceedings of the Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems, Yokohama, Japan, 8 May 2021. [CrossRef]
12. Elshan, E.; Zierau, N.; Engel, C.; Janson, A.; Leimeister, J.M. Understanding the design elements affecting user acceptance of intelligent agents: Past, present and future. *Inf. Syst. Front.* **2022**, *24*, 699–730. [CrossRef]
13. Bharadwaj, N.A.; Dubé, A.K.; Talwar, V.; Patitsas, E. How Parents and Children Interact with Digital Assistants in the Home: An Exploratory Study. In Proceedings of the Society for Research in Child Development, 2021, Virtual Biennial Meeting, 7–9 April 2021.
14. Li, T.J.-J. A Multi-Modal Intelligent Agent that Learns from Demonstrations and Natural Language Instructions. Ph.D. Thesis, Human-Computer Interaction Institute, School of Computer Science Carnegie Mellon University, Pittsburgh, PA, USA, 3 May 2021.
15. Li, T.J.-J.; Azaria, A.; Myers, B.A. SUGILITE: Creating multimodal smartphone automation by demonstration. In Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems, Denver, CO, USA, 6–11 May 2017. [CrossRef]
16. Li, T.J.-J.; Radensky, M.; Jia, J.; Singarajah, K.; Mitchell, T.M.; Myers, B.A. PUMICE: A Multi-Modal Agent that Learns Concepts and Conditionals from Natural Language and Demonstrations. In Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology, New Orleans, LA, USA, 17 October 2019; pp. 577–589. [CrossRef]
17. Alt, R.; Human, S.; Neumann, G. End-user empowerment in the digital age. In Proceedings of the 53rd Hawaii International Conference on System Sciences, Maui, HI, USA, 7–10 January 2020.
18. Lau, T. Why programming-by-demonstration systems fail: Lessons learned for usable ai. *AI Mag.* **2009**, *30*, 65. [CrossRef]
19. Barricelli, B.R.; Cassano, F.; Fogli, D.; Piccinno, A. End-user development, end-user programming and end-user software engineering: A systematic mapping study. *J. Syst. Softw.* **2019**, *149*, 101–137. [CrossRef]
20. Moussawi, S. User experiences with personal intelligent agents: A sensory, physical, functional and cognitive affordances view. In Proceedings of the 2018 ACM SIGMIS Conference on Computers and People Research, New York, NY, USA, 18–20 June 2018.
21. Følstad, A.; Brandtzaeg, P.B. Users' experiences with chatbots: Findings from a questionnaire study. *Qual. User Exp.* **2020**, *5*, 3. [CrossRef]
22. Abulfaraj, A.; Steele, A. Coherent Heuristic Evaluation (CoHE): Toward Increasing the Effectiveness of Heuristic Evaluation for Novice Evaluators. In *Design, User Experience, and Usability. Interaction Design. HCII 2020*; Lecture Notes in Computer Science; Marcus, A., Rosenzweig, E., Eds.; Springer: Cham, Swtzerland, 2020; Volume 12200. [CrossRef]
23. Maehigashi, A.; Miwa, K.; Kojima, K.; Terai, H. Development of a Usability Questionnaire for Automation Systems. In *Human-Computer Interaction. Theory, Design, Development and Practice. HCI 2016*; Lecture Notes in Computer Science; Kurosu, M., Ed.; Springer: Cham, Switzweland, 2016; Volume 9731. [CrossRef]
24. Qiu, S.; An, P.; Kang, K.; Hu, J.; Han, T.; Rauterberg, M. Investigating socially assistive systems from system design and evaluation: A systematic review. *Univers. Access Inf. Soc.* **2021**, *22*, 609–633. [CrossRef] [PubMed]
25. Quiñones, D.; Rusu, C. How to develop usability heuristics: A systematic literature review. *Comput. Stand. Interfaces* **2017**, *53*, 89–122. [CrossRef]
26. Page, M.J.; McKenzie, J.E.; Bossuyt, P.M.; Boutron, I.; Hoffmann, T.C.; Mulrow, C.D.; Shamseer, L.; Tetzlaff, J.M.; Akl, E.A.; Brennan, S.E. The PRISMA 2020 statement: An updated guideline for reporting systematic reviews. *Int. J. Surg.* **2021**, *88*, 105906. [CrossRef]
27. Modugno, F.; Myers, B.A. Pursuit: Graphically representing programs in a demonstrational visual shell. In Proceedings of the Conference Companion on Human Factors in Computing Systems, Boston, MA, USA, 24–28 April 1994.
28. Nielsen, J. Enhancing the explanatory power of usability heuristics. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Boston, MA, USA, 24–28 April 1994.
29. Lau, T.; Wolfman, S.A.; Domingos, P.; Weld, D.S. Learning repetitive text-editing procedures with SMARTedit. In *Your Wish Is My Command*; Elsevier: Amsterdam, The Netherlands, 2001; Volume XI, pp. 209–225. [CrossRef]
30. Chen, J.-H.; Weld, D.S. Recovering from errors during programming by demonstration. In Proceedings of the 13th International Conference on Intelligent User Interfaces, Gran Canaria Spain, 13–16 January 2008.
31. Bergman, L.; Castelli, V.; Lau, T.; Oblinger, D. DocWizards: A system for authoring follow-me documentation wizards. In Proceedings of the 18th Annual ACM Symposium on User Interface Software and Technology, Seattle, WA, USA, 23–26 October 2005.
32. Gulwani, S.; Hernández-Orallo, J.; Kitzelmann, E.; Muggleton, S.H.; Schmid, U.; Zorn, B. Inductive programming meets the real world. *Commun. ACM* **2015**, *58*, 90–99. [CrossRef]
33. Grabler, F.; Agrawala, M.; Li, W.; Dontcheva, M.; Igarashi, T. Generating photo manipulation tutorials by demonstration. In Proceedings of the ACM SIGGRAPH 2009 Papers, New Orleans, LA, USA, 3–7 August 2009; pp. 1–9.
34. Yeh, T.; Chang, T.-H.; Miller, R.C. Sikuli: Using GUI screenshots for search and automation. In Proceedings of the 22nd Annual ACM Symposium on User Interface Software and Technology, Bend, OR, USA, 29 October–2 November2009.
35. Vaithilingam, P.; Guo, P.J. Bespoke: Interactively synthesizing custom GUIs from command-line applications by demonstration. In Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology, New Orleans, LA, USA, 20–23 October 2019. [CrossRef]
36. Evensen, S.; Ge, C.; Demiralp, C. Ruler: Data programming by demonstration for document labeling. In Proceedings of the Findings of the Association for Computational Linguistics: EMNLP 2020, Online, 16–20 November 2020.

37. Intharah, T.; Turmukhambetov, D.; Brostow, G.J. Hilc: Domain-independent pbd system via computer vision and follow-up questions. *ACM Trans. Interact. Intell. Syst. (TiiS)* **2019**, *9*, 1–27. [CrossRef]
38. Kim, D.; Park, S.; Ko, J.; Ko, S.Y.; Lee, S.-J. X-droid: A quick and easy android prototyping framework with a single-app illusion. In Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology, New Orleans, LA, USA, 20–23 October 2019.
39. Hartmann, B.; Wu, L.; Collins, K.; Klemmer, S.R. Programming by a sample: Rapidly creating web applications with d. mix. In Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology, Newport, RI, USA, 7–10 October 2007.
40. Leshed, G.; Haber, E.M.; Matthews, T.; Lau, T. CoScripter: Automating & sharing how-to knowledge in the enterprise. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Florence Italy, 5–10 April 2008.
41. Lin, J.; Wong, J.; Nichols, J.; Cypher, A.; Lau, T.A. End-user programming of mashups with vegemite. In Proceedings of the 14th International Conference on Intelligent User Interfaces, Sanibel Island, FL, USA, 8–11 February 2009.
42. Barman, S.; Chasins, S.; Bodik, R.; Gulwani, S. Ringer: Web automation by demonstration. In Proceedings of the 2016 ACM SIGPLAN International Conference on Object-Oriented Programming, Systems, Languages, and Applications, Amsterdam, The Netherlands, 2–4 November 2016.
43. Chasins, S.E.; Mueller, M.; Bodik, R. Rousillon: Scraping distributed hierarchical web data. In Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology, Amsterdam, The Netherlands, 2–4 October 2018.
44. Rodrigues, A. Breaking barriers with assistive macros. In Proceedings of the 17th International ACM SIGACCESS Conference on Computers & Accessibility, Lisbon, Portugal, 26–28 October 2015.
45. Huang, T.-H.K.; Azaria, A.; Bigham, J.P. Instructablecrowd: Creating if-then rules via conversations with the crowd. In Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems, San Jose, CA, USA, 7–12 May 2016.
46. Sereshkeh, A.R.; Leung, G.; Perumal, K.; Phillips, C.; Zhang, M.; Fazly, A.; Mohomed, I. VASTA: A vision and language-assisted smartphone task automation system. In Proceedings of the 25th International Conference on Intelligent User Interfaces, Cagliari, Italy, 17–20 March 2020.
47. Yang, J.; Lam, M.S.; Landay, J.A. DoThisHere: Multimodal Interaction to Improve Cross-Application Tasks on Mobile Devices. In Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology, Virtual, 20–23 October 2020; pp. 35–44.
48. Krawiec, Ł.; Dudycz, H. Identification of heuristics for assessing the usability of websites of public administration units. In Proceedings of the 2019 Federated Conference on Computer Science and Information Systems (FedCSIS), Leipzig, Germany, 1–4 September 2019; IEEE: Piscataway, NJ, USA, 2019. [CrossRef]
49. Bouraghi, H.; Rezayi, S.; Amirazodi, S.; Nabovati, E.; Saeedi, S. Evaluating the usability of a national health information system with heuristic method. *J. Educ. Health Promot.* **2022**, *11*, 182. [CrossRef]
50. Faria Gomes, R.; Costa de Souza, M.d.F. Reprojecting a Fitness App Regarding Retention and Usability Using Nielsen's Heuristics. In *Design, User Experience, and Usability: Design for Diversity, Well-being, and Social Development. HCII 2021*; Lecture Notes in Computer Science; Soares, M.M., Rosenzweig, E., Marcus, A., Eds.; Springer: Cham, Switzerland, 2021. [CrossRef]
51. Figueroa, I.; Jiménez, C.; Allende-Cid, H.; Leger, P. Developing usability heuristics with PROMETHEUS: A case study in virtual learning environments. *Comput. Stand. Interfaces* **2019**, *65*, 132–142. [CrossRef]
52. Umar, M.M.; Bakhat, M.U.; Hassan, M. Mapping HCI Principals to Evaluate the Usability of Learning Applications for CCI User. *Int. J. Comput. Sci. Telecommun.* **2020**, *11*, 1–7.
53. Kumar, M.; Emory, J.; Choppella, V. Usability analysis of virtual labs. In Proceedings of the 2018 IEEE 18th International Conference on Advanced Learning Technologies (ICALT), Mumbai, India, 9–13 July 2018; IEEE: Piscataway, NJ, USA, 2018.
54. Ali, G.; Dida, M.A.; Sam, A.E. Heuristic Evaluation and Usability Testing of G-MoMo Applications. *J. Inf. Syst. Eng. Manag.* **2022**, *7*, 15751. [CrossRef]
55. Bashir, M.S.; Farooq, A.; Humayoun, S.R.; Iqbal, M.M.; Iqbal, M.J. EAUHHCAMA: Extending and Adapting Usability Heuristics for Healthcare Related Context-Aware Mobile Applications. *J. Med. Imaging Health Inform.* **2020**, *10*, 2345–2360. [CrossRef]
56. Limtrairut, P. Newly developed heuristics to evaluate m-learning application interface. In Proceedings of the 2020-5th International Conference on Information Technology (InCIT), Chonburi, Thailand, 21–22 October 2020; IEEE: Piscataway, NJ, USA, 2020.
57. Wei, Z.; Landay, J.A. Evaluating speech-based smart devices using new usability heuristics. *IEEE Pervasive Comput.* **2018**, *17*, 84–96. [CrossRef]
58. Hussain, A.; Barakat, M.M.; Zaaba, Z.F. Heuristic evaluation of stock exchange mobile application in Malaysia. *Int. J. Adv. Sci. Technol.* **2020**, *29*, 340–354.
59. Sancho Nascimento, L.; Zagalo, N.; Bezerra Martins, L. Challenges of developing a mobile game for children with Down Syndrome to test gestural interface. *Information* **2020**, *11*, 159. [CrossRef]
60. Muhanna, M.A.; Amro, R.N.; Qusef, A. Using a new set of heuristics in evaluating Arabic interfaces. *J. King Saud Univ.-Comput. Inf. Sci.* **2020**, *32*, 248–253. [CrossRef]
61. Derby, J.L.; Chaparro, B.S. The Development and Validation of an Augmented and Mixed Reality Usability Heuristic Checklist. In *Augmented and Mixed Reality: Design and Development. HCII 2022*; Lecture Notes in Computer Science; Chen, J.Y.C., Fragomeni, G., Eds.; Springer: Cham, Switzerland, 2022; Volume 13317. [CrossRef]
62. Da Costa, R.P.; Canedo, E.D.; De Sousa, R.T.; Albuquerque, R.D.O.; Villalba, L.J.G. Set of usability heuristics for quality assessment of mobile applications on smartphones. *IEEE Access* **2019**, *7*, 116145–116161. [CrossRef]

63. Saavedra, M.J.; Rusu, C.; Quiñones, D.; Roncagliolo, S. A Set of Usability and User eXperience Heuristics for Social Networks. In *Social Computing and Social Media. Design, Human Behavior and Analytics. HCII 2019*; Lecture Notes in Computer Science; Meiselwitz, G., Ed.; Springer: Cham, Switzerland, 2019; Volume 11578. [CrossRef]
64. Vieira, E.A.O.; Silveira, A.C.d.; Martins, R.X. Heuristic evaluation on usability of educational games: A systematic review. *Inform. Educ.* **2019**, *18*, 427–442. [CrossRef]
65. Abreu, C.A.; Rosa, J.C.S.; Matos, E.D. Usability Heuristics for Children Educational Mobile App. *Abakos* **2020**, *8*, 42–60. [CrossRef]
66. Marquez, J.O.; Meirelles, P.; da Silva, T.S. Towards Usability Heuristics for Interactive Web Maps. In Proceedings of the XX Brazilian Symposium on Human Factors in Computing Systems, Virtual Event Brazil, 18–22 October 2021.
67. Silvis, I.M.; Bothma, T.J.; de Beer, K.J. Evaluating the usability of the information architecture of academic library websites. *Libr. Hi Tech* **2019**, *37*, 566–590. [CrossRef]
68. Salah, M.S.; Jusoh, S.; Muhanna, M.A. The development of usability heuristics For Arabic m-commerce applications. In Proceedings of the 2019 IEEE Jordan International Joint Conference on Electrical Engineering and Information Technology (JEEIT), Amman, Jordan, 9–11 April 2019; IEEE: Piscataway, NJ, USA, 2019.
69. Ismail, H.; Khafaji, H.; Fasla, H.; Younis, A.R.; Harous, S. A cognitive style-based usability evaluation of Zoom and Teams for online lecturing activities. In Proceedings of the 2021 IEEE Global Engineering Education Conference (EDUCON), Vienna, Austria, 21–23 April 2021; IEEE: Piscataway, NJ, USA, 2021.
70. Zardari, B.A.; Hussain, Z.; Arain, A.A.; Rizvi, W.H.; Vighio, M.S. QUEST e-learning portal: Applying heuristic evaluation, usability testing and eye tracking. *Univers. Access Inf. Soc.* **2021**, *20*, 531–543. [CrossRef]
71. Tremoulet, P.D.; Shah, P.D.; Acosta, A.A.; Grant, C.W.; Kurtz, J.T.; Mounas, P.; Kirchhoff, M.; Wade, E. Usability of Electronic Health Record–Generated Discharge Summaries: Heuristic Evaluation. *J. Med. Internet Res.* **2021**, *23*, e25657. [CrossRef] [PubMed]
72. Anuar, N.N.; Othman, M.K. Development and validation of progressive web application usability heuristics (PWAUH). *Univ. Access Inf. Soc.* 2022. [CrossRef]
73. Kaya, A.; Gumussoy, C.A.; Ekmen, B.; Bayraktaroglu, A.E. Usability heuristics for the set-top box and TV interfaces. *Hum. Factors Ergon. Manuf. Serv. Ind.* **2021**, *31*, 270–290. [CrossRef]
74. Yanez-Gomez, R.; Font, J.L.; Cascado-Caballero, D.; Sevillano, J.-L. Heuristic usability evaluation on games: A modular approach. *Multimed. Tools Appl.* **2019**, *78*, 4937–4964. [CrossRef]
75. de Franceschi, V.D.; Fontoura, L.M.; Silva, M.A.R. Usability Heuristics for Tabletop Systems Design. *ICEIS* **2020**, *2*, 555–562.
76. Kumar, B.A.; Goundar, M.S. Usability heuristics for mobile learning applications. *Educ. Inf. Technol.* **2019**, *24*, 1819–1833. [CrossRef]
77. Marcilly, R.; Colliaux, J.; Robert, L.; Pelayo, S.; Beuscart, J.-B.; Rousselière, C.; Décaudin, B. Improving the usability and usefulness of computerized decision support systems for medication review by clinical pharmacists: A convergent, parallel evaluation. *Res. Soc. Adm. Pharm.* **2023**, *19*, 144–154. [CrossRef]
78. Sánchez-Adame, L.M.; Mendoza, S.; Urquiza, J.; Rodríguez, J.; Meneses-Viveros, A. Towards a set of heuristics for evaluating chatbots. *IEEE Lat. Am. Trans.* **2021**, *19*, 2037–2045. [CrossRef]
79. Samarakoon, S.; Weerasinghe, T.; Usoof, H. Usability Heuristics for Early Primary Children: A Case Study in Sri Lanka. In Proceedings of the 2021 International Conference on Computational Science and Computational Intelligence (CSCI), Las Vegas, NV, USA, 15–17 December 2021; IEEE: Piscataway, NJ, USA, 2021.
80. Huang, Z. Usability of tourism websites: A case study of heuristic evaluation. *New Rev. Hypermedia Multimed.* **2020**, *26*, 55–91. [CrossRef]
81. Viana, L.; Passos, L.; Oliveira, E.; Conte, T. Applying Usability Heuristics in the Context of Data Labeling Systems. In Proceedings of the XX Brazilian Symposium on Human Factors in Computing Systems, Virtual Event Brazil, 18–22 October 2021.
82. Eltalhi, S.; Kutrani, H.; Imsallim, R.; Elrfadi, M. The Usability of BenKids Mobile Learning App in Vocabulary Teaching for Preschool. *iJIM* **2021**, *15*, 5. [CrossRef]
83. Kim, M.J.; Schroeder, S.; Chan, S.; Hickerson, K.; Lee, Y.-C. Reviewing the User-Centered Design Process for a Comprehensive Gastroesophageal Reflux Disease (GERD) App. *Int. J. Environ. Res. Public Health* **2022**, *19*, 1128. [CrossRef] [PubMed]
84. Robson, R.S.; Sabahat, N. Heuristic based approach for usability evaluation of mobile games. In Proceedings of the 2020 International Conference on Computing, Electronics & Communications Engineering (iCCECE), Southend, UK, 17–18 August 2020; IEEE: Piscataway, NJ, USA, 2020.
85. Pyae, A. A usability evaluation of the Google Home with non-native English speakers using the system usability scale. *Int. J. Netw. Virtual Organ.* **2022**, *26*, 172–194. [CrossRef]
86. Kirkscey, R. Development and patient user experience evaluation of an mHealth informational app for osteoporosis. *Int. J. Hum.–Comput. Interact.* **2022**, *38*, 707–718. [CrossRef]
87. Bashir, M.S.; Farooq, A. EUHSA: Extending usability heuristics for smartphone application. *IEEE Access* **2019**, *7*, 100838–100859. [CrossRef]
88. Hasan, M.S.; Yu, H. Innovative developments in HCI and future trends. *Int. J. Autom. Comput.* **2017**, *14*, 10–20. [CrossRef]