JOISIE (Journal Of Information Systems And Informatics Engineering)
Vol. 9, No.1, Juni 2025, Hlm 218-227
Received: 28 April 2025 Revised: 25 Juni 2025, Accepted: 29 Juni 2025

p- ISSN: 2503-5304
e- ISSN: 2527-3116

IMPLEMENTATION OF A TELEGRAM BOT FOR REAL-TIME FLIGHT INFORMATION SERVICES AT SOEKARNO-HATTA INTERNATIONAL AIRPORT

Faiz Albanna[1)], Dhiani Dyahjatmayanti[2)], Ristiani[3)]
[1]Air Transport Management Study Program, College of Aerospace Technology, Yogyakarta, Indonesia
[2]Transport Management Study Program, College of Aerospace Technology, Yogyakarta, Indonesia
[3]Air Transport Management Study Program, College of Aerospace Technology, Yogyakarta, Indonesia
Email: [1]faiz@sttkd.ac.id, [2]dhiani.dyahjatmayanti@sttkd.ac.id,
[3]ristiani@sttkd.ac.id

Abstract

The increasing reliance on smartphones and instant messaging platforms has transformed how information is accessed, especially in public service environments such as airports. This study presents the development and evaluation of a Telegram bot designed to deliver real-time flight information at Soekarno-Hatta International Airport, supporting the Silent Airport initiative. Using the Waterfall development model, the bot was built through structured phases including data analysis, interface design, implementation, testing, and maintenance. The system integrates the AirLabs API and offers users personalized access to arrival, departure, and flight search features. A user experience survey involving 60 participants assessed four indicators using a 5-point Likert scale. The results showed high average scores: usefulness (4.38), ease of use (4.30), user satisfaction (4.30), and user acceptance (4.23). These scores indicate strong user approval, as they approach the maximum value of 5 on the Likert scale. Comparative analysis with traditional Flight Information Display Systems (FIDS) highlights the bot's strengths in accessibility, personalization, and mobile integration. Despite minor issues with real-time data accuracy, the Telegram bot effectively complements existing airport infrastructure and aligns with evolving digital user behavior. This research supports the potential of instant messaging platforms to enhance public information services within smart airport ecosystems.

Keywords: *flight information, instant messaging, Soekarno-Hatta Airport, Telegram bot, user experience.*

1. INTRODUCTION

The rapid advancement of communication technologies, particularly the proliferation of smartphones, has significantly influenced patterns of interaction and information dissemination. Instant messaging platforms have become integral across various environments, including academic and institutional settings, facilitating efficient coordination and access to services (Ling & Lai, 2016) Identified that the growing reliance on smartphones is largely driven by instant messaging and VoIP services (Azfar et al., 2016).

In modern airports, timely communication between systems and passengers is essential. As digital transformation accelerates in the aviation sector, there is a growing need for flexible and efficient communication tools that go beyond traditional display boards and loudspeakers. Among the many digital platforms available, Telegram stands out as one of the leading instant messaging applications. It offers features such as end-to-end encryption, cross-platform support, and an open API, which have made it increasingly popular for various types of system integrations, including bots. Telegram offers features such as end-to-end encryption, self-destructing messages, multi-data center infrastructure, cross-platform support, and an open API, which have made it increasingly popular for system integrations (Aljarih, 2025). A chatbot is an automated program that responds to user input in natural language, mimicking human interaction (Patel et al., 2019).

These bots have been utilized in a variety of domains, including educational tools, campus automation, and public service delivery (Heryandi, 2020). In Indonesia, several academic institutions have adopted Telegram bots for administrative and promotional tasks (Kurnaedi & Widodo, 2023). However, detailed historical background on Telegram's growth is less relevant to this study and will not be further elaborated.

In contrast to educational use cases, the application of messaging bots in the transportation sector particularly in airports remains underexplored, despite its significant potential. In airport environments implementing Silent Airport policies where public address announcements are deliberately reduced to control noise pollution the provision of timely and accurate flight information becomes critical (Heyes et al., 2021).

https://doi.org/10.35145/joisie.v9i1.4979

Traditional methods of delivering flight information such as flight information display systems (FIDS) and public address (PA) announcements often face critical limitations in terms of reach, personalization, and responsiveness. In busy airport terminals, passengers are frequently distracted by multiple stimuli, causing them to miss visual or auditory cues (Board, 2019).

This research addresses this gap by developing and implementing a Telegram bot designed to deliver real-time flight information at Soekarno-Hatta International Airport. The bot enables passengers to independently track flight schedules, gate changes, and delays, thereby enhancing information accessibility and supporting the Silent Airport initiative. Using the Waterfall development model, the bot is engineered for accuracy, timeliness, and ease of use.

2. RESEARCH METHODOLOGY

This research adopts the Waterfall model as the primary methodology for software development. The Waterfall approach is selected due to its structured, sequential process, which aligns well with the scope of this project developing a Telegram bot for flight information services. This model describes the phases of software development in a linear and sequential manner, with each phase only starting after the previous phase has been completed (Diansyah et al., 2023).

The two most common methodologies of software development to date are agile and waterfall. According to surveys, 71% of businesses prefer agile, whereas 51% of businesses utilize waterfall (Mishra & Alzoubi, 2023).The choice to adopt the Waterfall model over Agile model is grounded in the static and predictable nature of this project (Thesing et al., 2021). Unlike Agile, which encourages iterative changes and continuous user involvement, the Waterfall model offers a practical framework for projects where requirements are known upfront and unlikely to change significantly.

Table 1. Comparison of the Models

Criteria	*Waterfall*	*Agile*	*Result*
Requirement Stability	Clearly defined from the beginning	Frequently evolving	Waterfall
Development Process	Linear and phase-based	Iterative and incremental	Waterfall
Project Scope and Complexity	Small to medium scale with limited modules	Large-scale or dynamic projects	Waterfall
User Interaction	Minimal after initial analysis	High-frequency collaboration required	Waterfall
Documentation Level	Comprehensive at each stage	Often lightweight and adaptive	Waterfall

The Waterfall model ensures a practical and easy-to-understand workflow for both developers and evaluators (Saravanos & Curinga, 2023). Its clarity makes it ideal for academic and applied research contexts involving structured development processes. The development process begins with the analysis of relevant data to support the bot's database. This stage is followed by interface design, translation of the design into a programming language, implementation and testing, and ultimately maintenance, which encompasses modifications based on user feedback or system errors identified after deployment.

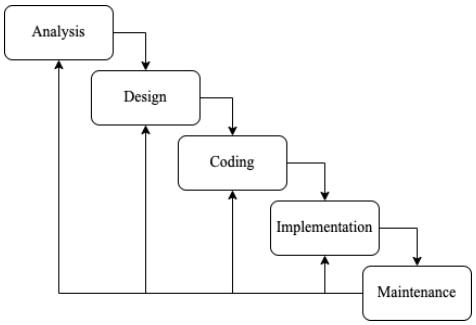

Figure 1. The Waterfall Model

Phases of the Waterfall Model:

a. Analysis
   In this phase, supporting data used to construct the Telegram bot's database is analyzed. This data is primarily derived from frequently asked questions posed by prospective users.
b. Design
   In this phase, the user interface is designed based on the results of the analysis. The data collected is organized into a structured navigation system.
c. Coding

https://doi.org/10.35145/joisie.v9i1.4979

This phase translates the interface design into a programming language. It results in a Telegram bot navigation system that aligns with the previously designed interface.

d. Implementation
Implementation includes deploying the bot and conducting test runs to minimize errors and ensure that the output matches the initial design specifications.

e. Maintenance
The final phase involves maintaining the system after deployment. Adjustments may be required due to undetected errors during testing or user feedback indicating performance issues.

The system architecture of the Telegram bot is designed to support efficient communication between users and real-time flight information services. The system consists of five main components: the Telegram user, the Telegram Bot, backend server, MySQL database, and an external flight data API.

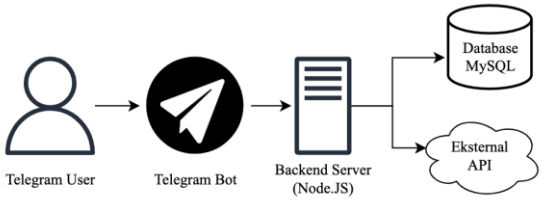

Figure 2. System Architecture

As illustrated in Figure 2, the system operates as follows:

a. Telegram User: The end user accesses the bot through the Telegram mobile or desktop application, sending queries regarding flight information.
b. Telegram Bot: The Telegram application forwards the user's input to the Telegram Bot API, which acts as the communication bridge between the user interface and the backend system.
c. Backend Server (Node.js): The bot logic is hosted on a backend server built using Node.js, which processes user queries, manages session flow, and coordinates data retrieval.
d. MySQL Database: The server queries the MySQL database for static or frequently used data, such as FAQs, gate numbers, and airport terminal mappings. This reduces dependence on external APIs for repeat requests.
e. External API: For dynamic or real-time data, such as flight departure/arrival times and gate updates, the backend interacts with an external API.

Data flows in a unidirectional pattern, starting from the user and moving through the Telegram Bot to the backend. The backend then fetches relevant data from either the database or the external API before returning the response via the Telegram application. This architecture ensures high maintainability, scalability, and clear separation of concerns between interface, logic, and data layers.

The system is developed using a set of technologies that support real-time communication, data storage, and integration with flight data APIs. These are summarized in Table 2.

Table 2. Summarizes the key technologies used in the bot development lifecycle.

Component	***Technology/Tools***	***Purpose***
Programming Language	Node.js (JavaScript runtime)	Backend logic and API integration
Database	MySQL	Storing flight data and user interaction logs
Bot Platform	Telegram Bot API	User interaction and message routing
External API	AirLabs.co	Real-time flight schedule retrieval
Development Environment	Visual Studio Code	Development and debugging
Hosting	STTKD Data Center	Hosting the bot and ensuring 24/7 availability

The scope of this research is limited to flight information services at Soekarno-Hatta International Airport in Tangerang (CGK), particularly focusing on data concerning departure and arrival gates for passengers.

3. RESULTS AND DISCUSSION

The primary objective of the analysis conducted in this study was to identify the necessary data to be integrated into the Telegram bot's database. The required flight data were collected through the AirLabs API. Upon retrieving the data, a table design was developed to define the structure of the bot's user interface and navigation system.

https://doi.org/10.35145/joisie.v9i1.4979

Table 3. Telegram Bot Interface Design

Category	*Sub Category*	*Command*	*Interaction*	*Output*
Start [start]	Arrivals	/arrival	*Multi*	Teks & Navigasi
	Departures	/departure	*Multi*	Teks & Navigasi
	Flight Search	/search	*Multi*	Teks & Navigasi
Arrivals [arrival]	Arrivals	/arrival	*Multi*	Teks & Navigasi
Departures [departure]	Departures	/departure	*Multi*	Teks & Navigasi
Flight Search [search]	Flight Number	/search_flight	*Single*	Teks
	Airline	/search_airline	*Multi*	Teks & Navigasi

Based on the design, four main categories were established:

a. Start, subdivided into three subcategories.
b. Arrivals, with one subcategory.
c. Departures, with one subcategory.
d. Flight Search, divided into two subcategories.

The interaction types were classified as multi-message, which involves interactions with multiple navigational elements, and single-message, which includes only one navigational interaction. Following the interface layout, the workflow of the Telegram bot was modeled.

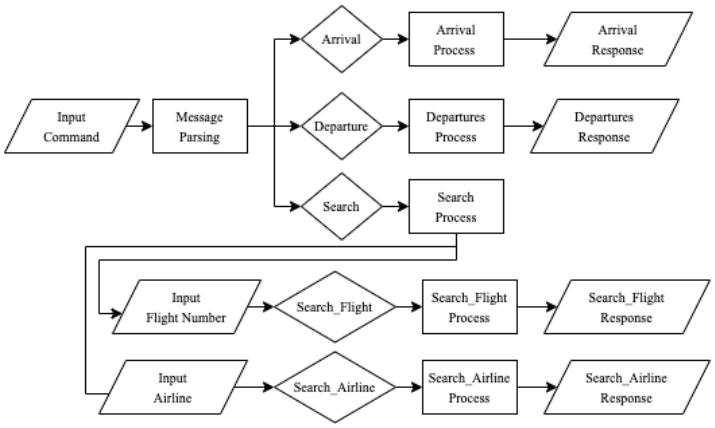

Figure 3. The Telegram Bot Workflow

Each bot process corresponds to the command selected by the user, resulting in different processing flows and responses. After establishing the interface and process flow, the bot was registered using the BotFather account to generate a bot name and API token, which facilitate communication between the bot and the server.

Table 4. Bot Registration Details

Identity	**Description**
Username	@soettaflighttrackerbot
Token API	7861642259:AAGvf2W1m5Tu1qjNROlphdeKDGmU5pOnc88
Address	http://t.me/soettaflighttrackerbot
Description	Soetta Flight Tracker helps users track real-time flight schedules at Soekarno-Hatta International Airport.
Botpic	Has a botpic
About	Monitor real-time flight schedules at Soekarno-Hatta International Airport.

Once registration was complete, the bot was programmed using JavaScript, and its development followed the previously designed flow. Below is an example of the implemented code for token authentication and response interaction:

https://doi.org/10.35145/joisie.v9i1.4979

```
const TelegramBot = require('node-telegram-bot-api');
const axios = require('axios');
const mysql = require('mysql2/promise');

const token = '7861642259:AAGvf2W1m5Tu1*********';
const bot = new TelegramBot(token, { polling: true,
request: {
    agentOptions:{
        keepAlive: true,
        family: 4}
}});
```

Figure 4. Implementation of Token API

The coding phase utilized various functions, including callbacks, to process and respond to user messages interacting with the bot.

```
bot.onText(/\/start/, (msg) => {
  const chatId = msg.chat.id;
  const username = msg.from.username || msg.from.first_name;

  const welcomeMessage = `Selamat datang di Flight Tracker, ${username}! \nBot
ini bertujuan untuk mengetahui jadwal penerbangan yang ada di Jakarta Soekarno
Hatta International (CGK). Pilih opsi berikut:`;
  const options = {
    reply_markup: {
      inline_keyboard: [
        [
          { text: 'Kedatangan', callback_data: 'Arrivals_0' },
          { text: 'Keberangkatan', callback_data: 'Departures_0' },
        ],
        [
          { text: 'Pencarian Penerbangan', callback_data: 'search' },],],},};
  bot.sendMessage(chatId, welcomeMessage, options);});
```

Figure 5. Implementation of Callback Message Responses

The next phase involved testing the Telegram bot to assess user experience and functionality in providing flight information.

4.1. ACCESSING THE TELEGRAM BOT

Telegram bots are features within the Telegram platform that perform specific automated functions in response to user commands. In this study, a Telegram bot is utilized to provide real-time flight information at Soekarno-Hatta International Airport.

Instructions for Accessing the Soetta Flight Tracker Bot:

a. Open the Telegram application, search for the bot by its name “Soetta Flight Tracker”, or directly access it via the following link http://t.me/soettaflighttrackerbot.

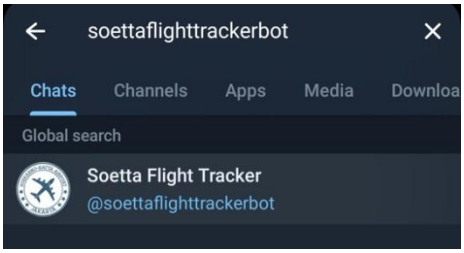

Figure 6. Bot Search Interface

https://doi.org/10.35145/joisie.v9i1.4979

b. Click Start to initiate the bot

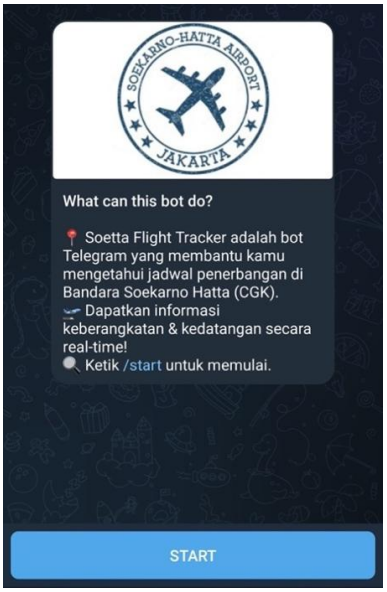

Figure 7. Start Menu

c. After clicking "Start," users can choose from three main menu options: Arrivals, Departures, and Flight Search. These menus are also accessible through inline buttons in the chat interface, which correspond to specific bot commands.

Figure 8. Bot Menu Interface

Functionality Overview of Bot Menus:

a. Arrival Menu
   Displays flight information including airline name, departure time from the origin airport (e.g., DPS), estimated arrival time at CGK, arrival terminal and gate, and the current flight status.
b. Departures Menu
   Display the airline, departure time from Soekarno-Hatta (CGK), estimated arrival time at the destination (e.g., DPS), arrival terminal and gate at the destination, and flight status.
c. Flight Search Menu
   Enables users to search for flights by flight number or airline code. The search results include the airline, departure and arrival airports, times, and flight status.:
   1) Flight Search by Number
      Select the "Number" option, input the flight number, and the bot will return detailed information for that flight on the current date.

https://doi.org/10.35145/joisie.v9i1.4979

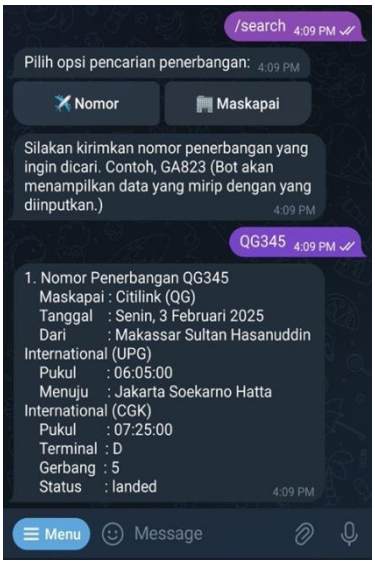

Figure 9. Flight Number Search Menu

2) Flight Search by Airline Code
Select the "Airline" option, input the airline code, and the bot will return a list of flights operated by that airline for the current date.

Figure 10. Airline Code Search

3) Additional Navigation
Commands can also be entered manually to substitute the function of the navigation buttons.

Figure.11. Additional Navigation Menu

4.2. SYSTEM IMPLEMENTATION OUTCOME

The Telegram bot Soetta Flight Tracker was successfully developed and deployed using the Waterfall model, which allowed for a linear and structured development process. The implementation began with the design of the database structure and interface flow, followed by backend programming using Node.js, integration with the AirLabs flight data API, and testing through Telegram's user interface.

https://doi.org/10.35145/joisie.v9i1.4979

The bot interface includes three main functions: Arrivals, Departures, and Flight Search. Each feature returns real-time data including flight number, terminal, gate, and status updates. The menu interface was designed with multi-message interactions and inline navigation to ensure user accessibility.

Figures and interface tables demonstrated a fully functional bot system, with features accessible via both manual commands and interactive buttons. The backend integration supports dynamic data retrieval, and a fallback database ensures performance stability even in the event of temporary API downtime.

4.3. USER EVALUATION RESULTS

A user experience survey was conducted to assess four key indicators: usefulness, ease of use, user satisfaction, and user acceptance. A total of 60 participants completed a questionnaire based on a 5-point Likert scale. The following are the summarized results:

a. Usefulness: Average score of 4.38. Users considered the bot highly relevant for flight tracking, especially gate and terminal information.
b. Ease of use: Average score of 4.30. Users found the bot intuitive and easy to operate.
c. User satisfaction: Average score of 4.30. Most participants were satisfied with the bot's responsiveness and performance.
d. User acceptance: Average score of 4.23. The majority stated that they would continue using the bot and would recommend it to others.

Although most responses were positive, a few users reported inconsistencies in flight data during the early implementation stage, supporting the need for improved real-time data access. These findings indicate that the Telegram bot generally met user expectations in terms of functionality and usability, while also highlighting areas for technical refinement to enhance reliability in future iterations.

4.4. COMPARATIVE DISCUSSION

This section presents a comparative analysis between Telegram bot and the conventional Flight Information Display System (FIDS), with emphasis on four critical dimensions: flexibility, information personalization, real-time accuracy, and user experience. The objective is to evaluate the extent to which a mobile-based information system can complement or enhance the functionalities of traditional airport infrastructure devices.

a. Personalization information: The bot enables customized, query-based information retrieval, while FIDS displays a generalized list of flights, which may be difficult to navigate quickly — especially during peak hours.
b. Real-time Accuracy: FIDS may be more tightly integrated with airport control systems, thus providing slightly more accurate and instantly updated data, especially in the case of last-minute gate changes or delays.
c. User Experience: The bot provides a more user-friendly, conversational experience, especially helpful for novice travellers. FIDS, while functional, is passive and less adaptive to individual needs.

Based on the dimensions assessed, the Telegram bot emerges as a viable and effective complement to existing FIDS infrastructure. While FIDS remains essential for on-site flight information delivery, the bot offers enhanced accessibility, individual-level customization, and mobile integration that align with contemporary digital user behavior. Its contribution is especially significant in supporting Silent Airport policies, where reliance on audio announcements is minimized, and personalized digital services are prioritized.

The findings suggest that the integration of instant messaging platforms into public information systems represents a promising direction for improving user engagement, operational efficiency, and passenger autonomy in the context of smart airport ecosystems.

4. CONCLUSION

The evaluation of the Telegram bot reveals strong user acceptance and high perceived usefulness, with average Likert-scale ratings consistently above 4.2 across multiple indicators. Users reported that the bot is easy to use, saves time, and provides relevant information tailored to their needs at Soekarno-Hatta International Airport.

https://doi.org/10.35145/joisie.v9i1.4979

Compared to Flight Information Display Systems (FIDS), the Telegram bot demonstrates significant advantages in accessibility, personalization, and user engagement. Unlike FIDS, which is limited to on-site access and static information, the bot offers real-time, mobile access that can be used before, during, or after travel.

While the system excels in usability and user satisfaction, limitations related to real-time data dependency and coverage scope suggest areas for future enhancement. Integrating the bot more directly with airport data systems and expanding its functionality across more airports may further improve performance and reliability.

In conclusion, the Telegram bot represents an effective and contextually appropriate digital innovation for improving air travel information access in Indonesia. It complements existing systems like FIDS while meeting the expectations of mobile-first, digitally engaged users.

5. ACKNOWLEDGMENT

The authors would like to express their sincere gratitude to the Center for Research and Community Service (P3M) of the College of Aviation Technology (STTKD) Yogyakarta, for funding and supporting this research project.

6. REFERENCES

Aljarih, F. (2025). A Comparative Study of WhatsApp and Telegram in Enhancing English Language Learning: A Case Study at the English Department, University of Benghazi, Suluq Campus. *Afaq Journal for Humanities and Applied Studies*, 266–279. https://doi.org/10.37376/ajhas.vi3.7248

Azfar, A., Choo, K.-K. R., & Liu, L. (2016). An Android Communication App Forensic Taxonomy. *Journal of Forensic Sciences*, *61*(5), 1337–1350. https://doi.org/10.1111/1556-4029.13164

Board, T. R. N. A. of S. E. and M. (2019). *Communication Strategies for Airport Passenger Access and Mobility* (W. W. Shu Cole Haoai Zhao Yan Zhang Indiana University & O. D. O. Laurel Van Horn, Eds.). The National Academies Press. https://doi.org/10.17226/25640

Diansyah, A., Rahman, M., Handayani, R., Cahyo, D., & Utami, E. (2023). Comparative Analysis of Software Development Lifecycle Methods in Software Development: A Systematic Literature Review. *International Journal of Advances in Data and Information Systems*, *4*, 97–106. https://doi.org/10.25008/ijadis.v4i2.1295

Heryandi, A. (2020). Developing Chatbot For Academic Record Monitoring in Higher Education Institution. *IOP Conference Series: Materials Science and Engineering*, *879*(1), 012049. https://doi.org/10.1088/1757-899X/879/1/012049

Heyes, G., Hooper, P., Raje, F., Flindell, I., Dimitriu, D., Galatioto, F., Burtea, N. E., Ohlenforst, B., & Konovalova, O. (2021). The Role of Communication and Engagement in Airport Noise Management. *Sustainability*, *13*(11). https://doi.org/10.3390/su13116088

Kurnaedi, D., & Widodo, A. D. (2023). Implementation of Telegram Chatbot as an Effective Communication Means at SMK PGRI 1 Tangerang. *Bit-Tech*, *6*(2), 183–189. https://doi.org/10.32877/bt.v6i2.1054

Ling, R., & Lai, C.-H. (2016). Microcoordination 2.0: Social Coordination in the Age of Smartphones and Messaging Apps. *Journal of Communication*, *66*(5), 834–856. https://doi.org/10.1111/jcom.12251

Mishra, A., & Alzoubi, Y. I. (2023). Structured software development versus agile software development: a comparative analysis. *International Journal of System Assurance Engineering and Management*, *14*(4), 1504–1522. https://doi.org/10.1007/s13198-023-01958-5

Patel, F., Thakore, R., Nandwani, I., & Bharti, S. K. (2019). Combating Depression in Students using an Intelligent ChatBot: A Cognitive Behavioral Therapy. *2019 IEEE 16th India Council International Conference (INDICON)*, 1–4. https://doi.org/10.1109/INDICON47234.2019.9030346

Saravanos, A., & Curinga, M. X. (2023). Simulating the Software Development Lifecycle: The Waterfall Model. *Applied System Innovation*, *6*(6). https://doi.org/10.3390/asi6060108

Thesing, T., Feldmann, C., & Burchardt, M. (2021). Agile versus Waterfall Project Management: Decision Model for Selecting the Appropriate Approach to a Project. *Procedia Computer Science*, *181*, 746–756. https://doi.org/10.1016/j.procs.2021.01.227

https://doi.org/10.35145/joisie.v9i1.4979