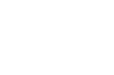
61



**Chapter III**

**Methodology**

In this chapter, the developer will explore the development process behind the "Application Approval for Vehicle Sticker". The developer utilized various software programs and development tools to create the system, with assistance and guidance as needed. We will provide a detailed overview of the software and hardware specifications used in the project.

# Project Design

The software project, entitled “Application Approval for Vehicle Sticker”, includes the development of both the front-end and back-end components using HTML, CSS, JavaScript, ReactJS, Django, and other Web Programming languages & Frameworks that will possibly be used for the development of this project. For this Web-Based System specifically, Visual Studio Code was used as the coding editor.

The program intends to provide a user-friendly interface, easy navigation through a personal access control dashboard, and data management. SQLite was used to create the database for the back-end. The said software is being improved in terms of usability and functionality to ensure effective data management.

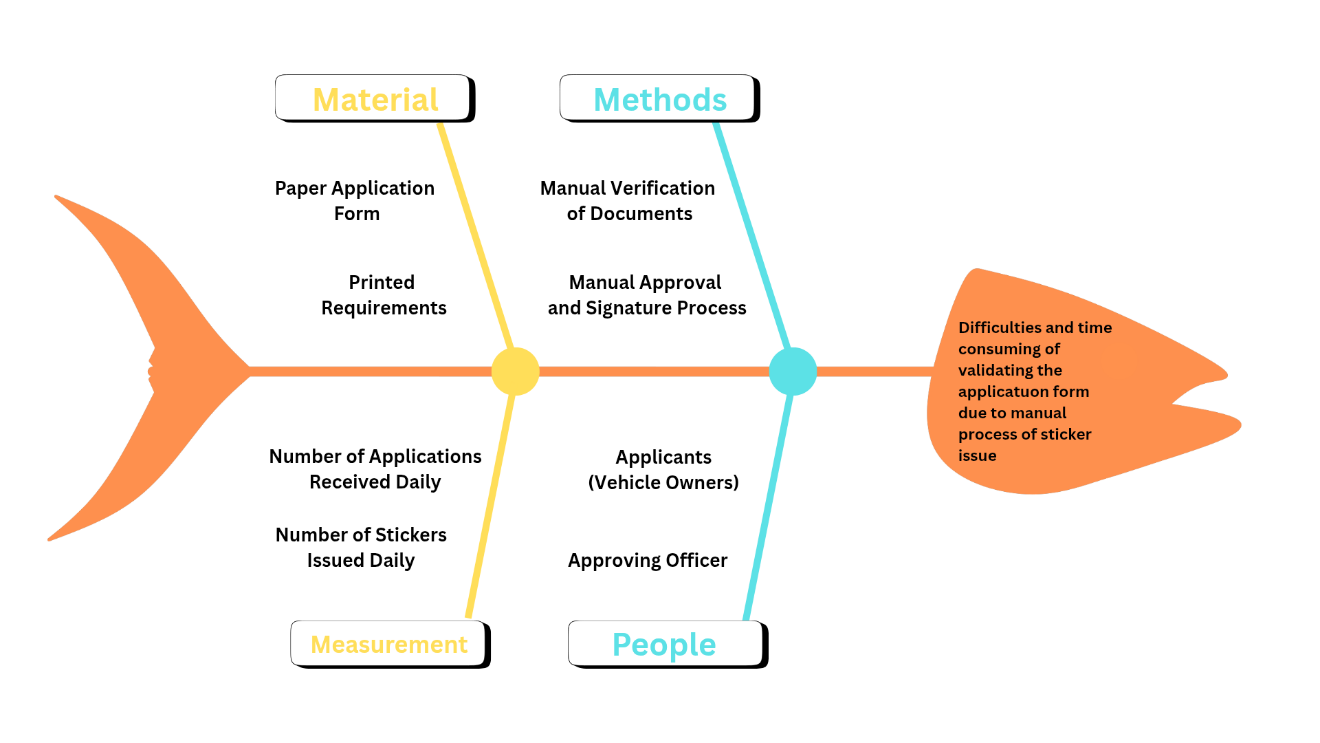
Application approval is envisioned to process the online submission of application requirements for the vehicle sticker by the employee from the campus security services or the production and commercialization of the institution. This will eliminate the long queue of applicants, validating requirements, and scheduling issues. The system’s structure is developed using ReactJS as frontend of the web development, with CSS handling design elements and JavaScript and Django as backend that handle all the functions and data processing.

The presentation and discussion of the software project will utilize a range of diagrams and illustrations to effectively depict the system's operations and showcase its overall design:

**Fishbone Diagram.** The Fishbone diagram is a visual tool used to identify and analyze the causes of specific events. it is applied as an innovative method to explore the sources of innovation, particularly General Purpose Technologies (GPTs). The research first identifies key drivers of GPTs, such as greater democratization and substantial investment in research and development. These drivers are then structured within Fishbone diagrams for a systematic technological analysis. The diagram is applied to specific GPTs, including the steam engine and information and communication technologies (ICTs), to illustrate its use. The Fishbone diagram proves to be a valuable and adaptable tool for visually representing technological advancements and forecasting transformative innovations in society (Coccia, 2020). This versatile tool acts as an essential instrument for researchers who need to understand complex problems and achieve solution effectiveness and determine fundamental sources. Its adaptable nature enables the tool to serve multiple sectors and academia while improving both solution-making and problem-solving activities.

# Figure 1

## Fishbone Diagram



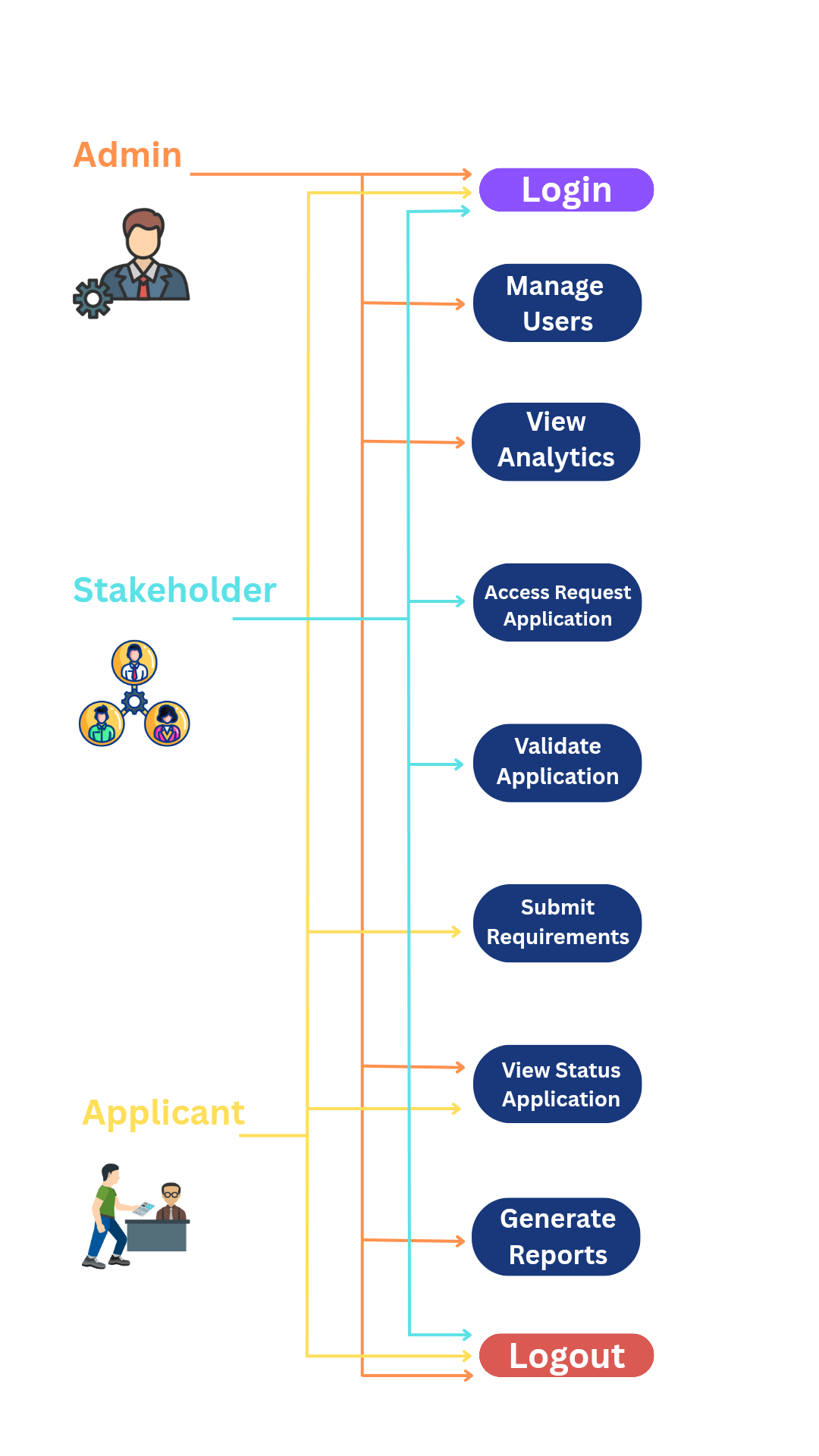
The fishbone diagram examines the vehicle sticker application challenges through material, methods, measurement, and people categories, which break down possible causes for analysis. Applicants require essential documents, including paper-based application forms as well as printed requirements according to the material category. The processing delays stem from missing or incomplete materials as well as poor-quality materials. The current methods section includes manual document verification upon manual approval and signature process, which produces lengthy processing times and is subject to human errors, particularly during busy operating periods. The process efficiency is measured through daily application volume counts and daily sticker distribution data. Huge differences between these two metrics signal potential operational problems, especially delays along with backlogs. Among people involved in the vehicle identification process, the approving officer with the person in charge, and the vehicle owners operate as essential components. The failure to communicate properly or handling too many applications results in slower and less precise application processing. The elements mentioned above construct our fishbone diagram which enables us to locate and handle the base causes behind sticker application system inefficiencies which is identified as a significant problem to address through the development of an automated system for online submission and approval of forms.

**Use Case Diagram.** A use case diagram, a type of Unified Modeling Language (UML), is commonly used to describe software functionality, but manual creation can lead to errors. Previous studies addressed this by generating diagrams from text inputs like user stories, though they mainly focused on actor-use case associations and used outdated UML editors. This research develops web-based software that generates use case diagrams from software requirement descriptions, including user stories and detailed relationship formats. It uses PlantUML as the editor, built with React and TypeScript, translating user input into PlantUML syntax through a sentence pattern approach. The software encodes the syntax to create use case diagram images and exports documents containing the requirements, syntax, and diagrams. Testing confirmed that the software accurately generates use case diagrams from user-provided functional descriptions. (Fathiyah & Widyani, 2024).

In summary, A use case diagram is essential because it visually represents a system's functionality from the user's perspective, helping all stakeholders understand the system's behavior. It improves communication among developers, designers, and business analysts by aligning the system with user requirements and business goals. Use case diagrams also define the system's scope, identify key behaviors and interactions, and serve as a basis for testing. They are valuable for documentation and future system updates, ensuring that all required functionality is covered and tested.

**Figure 2**

*Use Case Diagram*

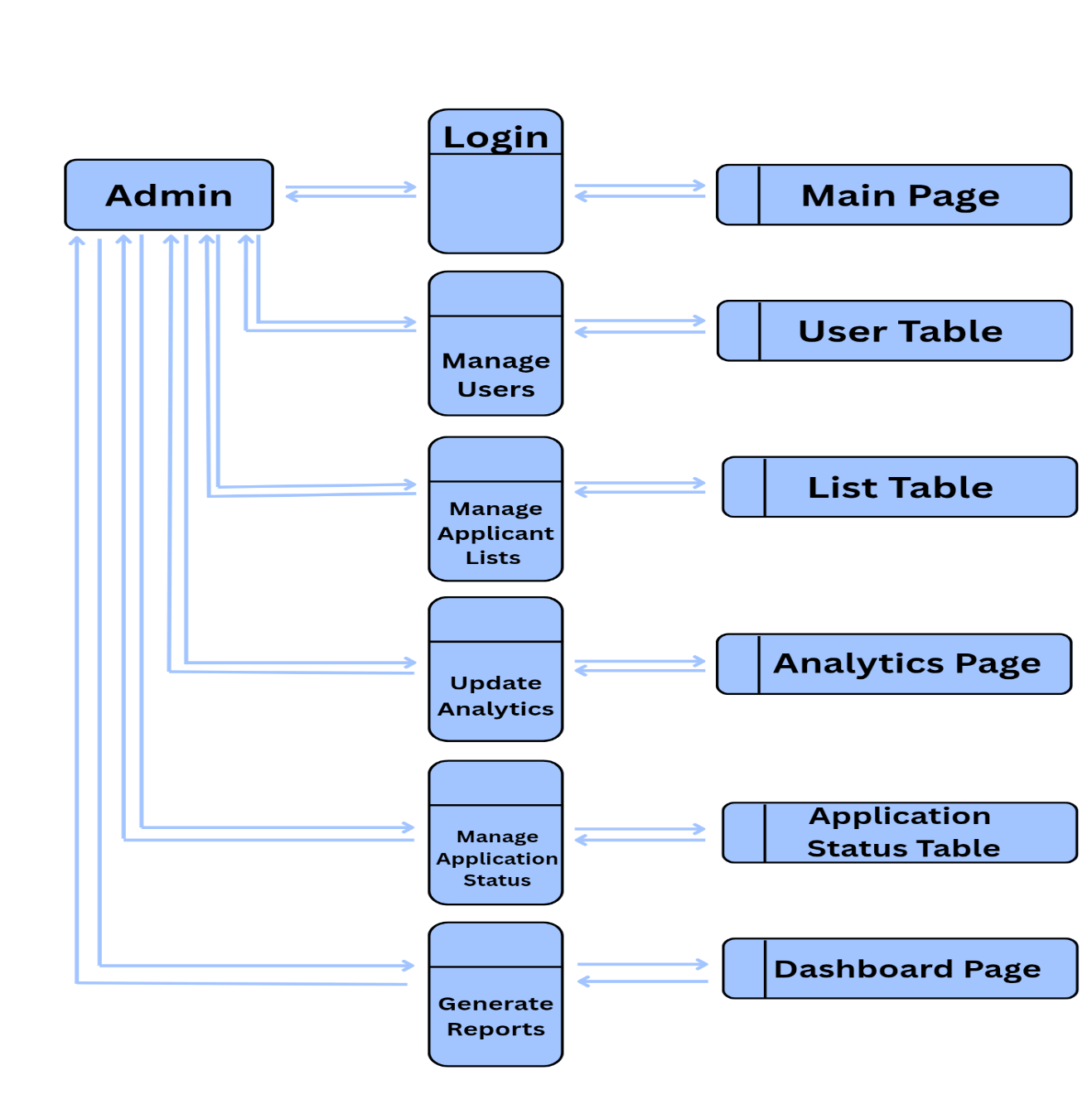
+

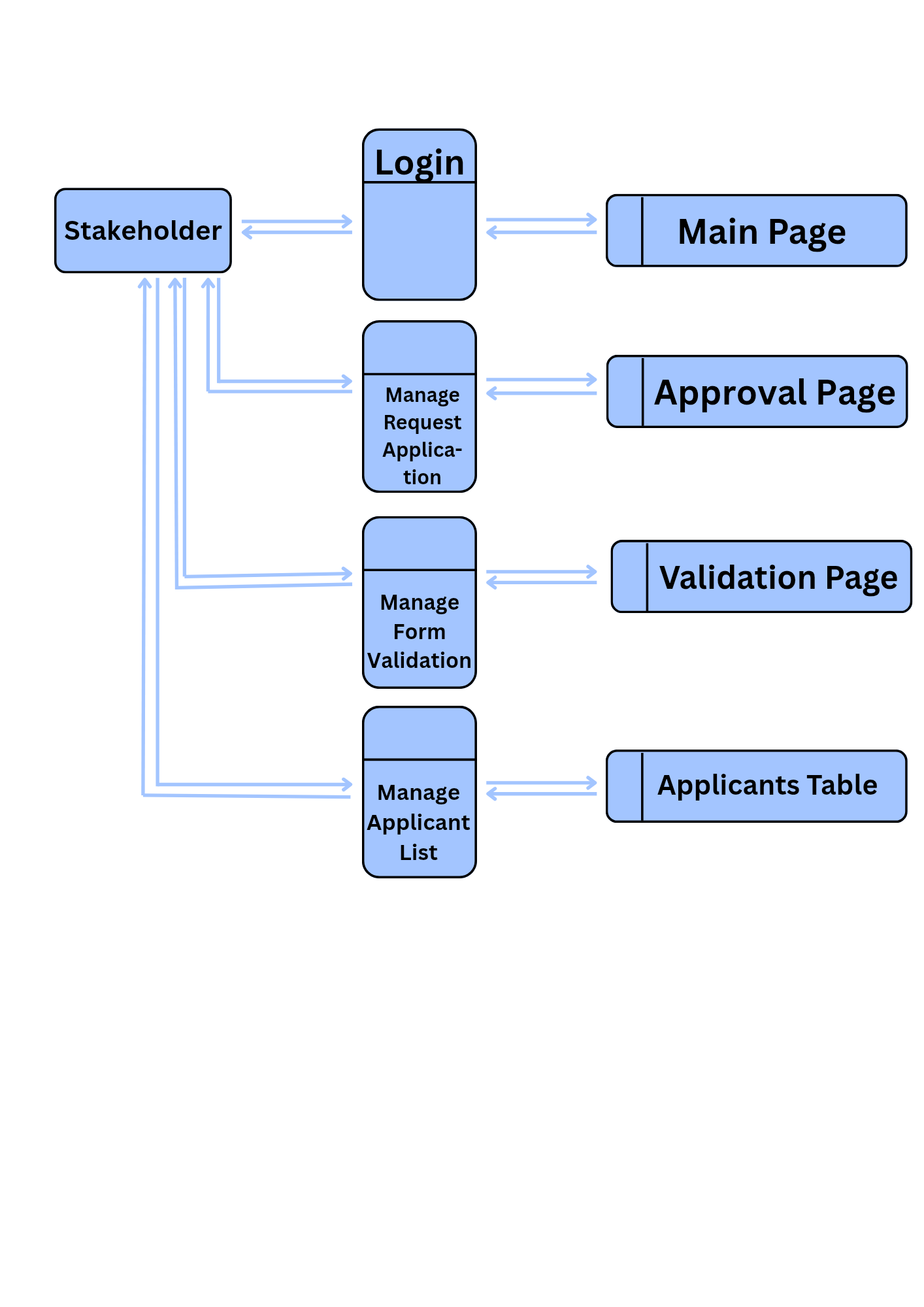
The administrator of the system can log in and out of the system, Furthermore, the administrator can manage users, view analytics, view application status, and generate reports. The stakeholder can log in and out, access the request application, and validate the application. Lastly, applicants can log in and out, submit requirements, and view application status.

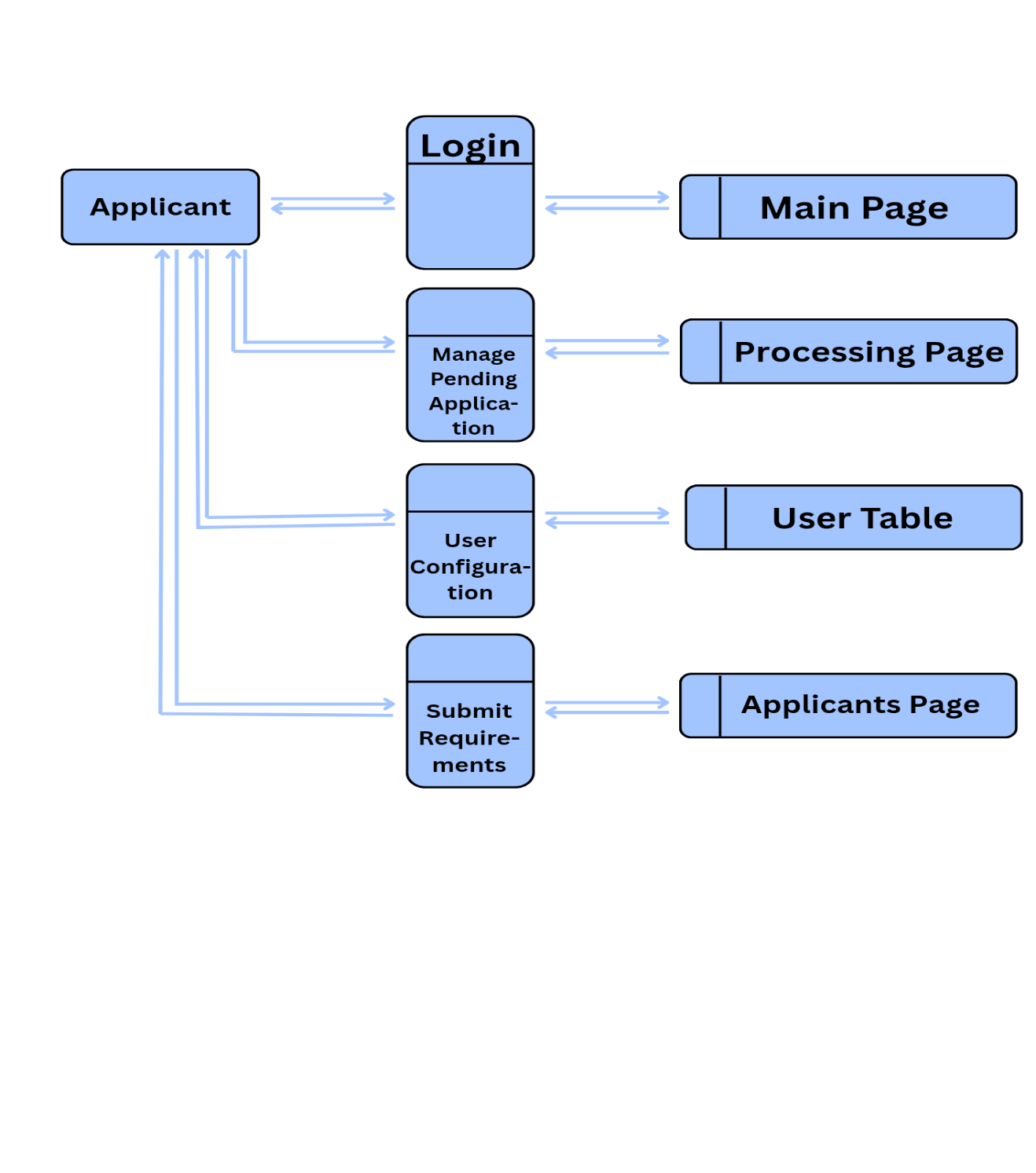
**Data Flow Diagram.** A data flow diagram (DFD) serves as a visual tool that depicts the movement of data within a system. It provides a structured representation of the processes that manage data, the data stores that retain information, and the external entities that interact with the system. By utilizing standardized symbols and directional arrows, DFDs facilitate clear communication of how data is processed and transferred. They play a vital role in contemporary software development by supporting the design process, enhancing system comprehension, and identifying potential security and privacy vulnerabilities (Herwanto, 2024).

# Figure 3

## Data Flow Diagram







The model structure allows only the admin users can access the System's Database. The admin has full modification privileges, while the Stakeholder user has similar access except for application form management, and is responsible for managing the request application. As for applicants, they can only access the applicant dashboard, view application status, submit requirements, and user configuration. The data flow modeling employs a limited number of constructs and follows straightforward and easily understandable rules.

# Technical Requirements

**Hardware.** Hardware constitutes the tangible elements of a computer system, which include machine parts within its structure as well as external connection tools. Data storage functions, along with input processing and output delivery, are managed within the computer hardware system. User interaction with the computer becomes enhanced through external peripherals that include monitors and printers to achieve better usability and operational performance.

The hardware specification used in the development of the proposed system is stated in detail:

**Laptop Model:** HP Victus

**Windows Version:** Windows 11 Home Single Language

**Processor:** AMD Ryzen 5 5600H with Radeon Graphics 3.30 GHz

# Installed Memory (RAM): 16.0 GB (15.3 GB usable)

**System Type:** 64-bit operating system, x64-based processor

**Software.** According to IEEE Xplore, software refers to a collection of programs, procedures, and routines associated with the operation of a computer system. It includes not only the application programs that perform specific tasks for users but also the system software that manages and controls the computer hardware. Software plays a critical role in enabling computers to perform complex computations, manage data, and provide services across various industries. IEEE Xplore emphasizes that software development involves activities such as requirements gathering, system design, programming, testing, and maintenance. Furthermore, software can be deployed through traditional installations, digital downloads, or cloud-based platforms, supporting a wide range of user needs from business operations to scientific research.

Software development utilizes a structured and precise set of descriptions to delineate the software employed. These descriptions are as follows:

*Visual Studio Code.* Visual Studio Code, developed by Microsoft, is a highly versatile and efficient source-code editor that is widely used by developers around the world. It provides a comprehensive set of features that enhance the coding experience across a variety of programming domains, including web development, mobile app creation, and cloud computing. Notable features such as IntelliSense, which offers intelligent code completion and syntax highlighting, help streamline the coding process while ensuring accuracy. Visual Studio Code also includes powerful debugging capabilities, allowing developers to identify and fix errors directly within the editor. One of its standout qualities is its extensibility, with a vast collection of extensions that enable customization for different programming languages, frameworks, and workflows. The editor also integrates a built-in terminal, enabling developers to run command-line tasks seamlessly. Additionally, native support for version control systems like Git, along with extensive customization options, ensures that developers can create a personalized coding environment. Overall, Visual Studio Code’s popularity and praise stem from its unmatched combination of performance, versatility, and a rich feature set, making it an essential tool for modern software development (Microsoft Corporation, 2021).

The SQLite is a widely used, open-source, relational database management system (RDBMS) that is embedded directly into applications, requiring minimal configuration. Unlike traditional server-based databases, SQLite operates as a lightweight, serverless, self-contained database engine. It stores data in a single file, making it highly portable and easy to set up. SQLite uses SQL (Structured Query Language) for querying and managing the database, providing developers with a familiar interface for handling relational data. It is designed for low-to-medium scale applications and is commonly used in mobile apps, embedded systems, and small-to-medium desktop applications. SQLite is highly efficient in terms of storage and performance, particularly when dealing with smaller datasets. However, it may face limitations in handling very large-scale applications with high concurrency demands, as it does not support full-scale multi-user access or complex distributed systems. Despite these limitations, SQLite's simplicity, speed, and minimal overhead make it an ideal solution for many embedded systems, local storage, and single-user applications (SQLite Consortium, n.d.).

The proposed system will rely on this platform as its server, and all functionalities will be accessible only upon its launch.

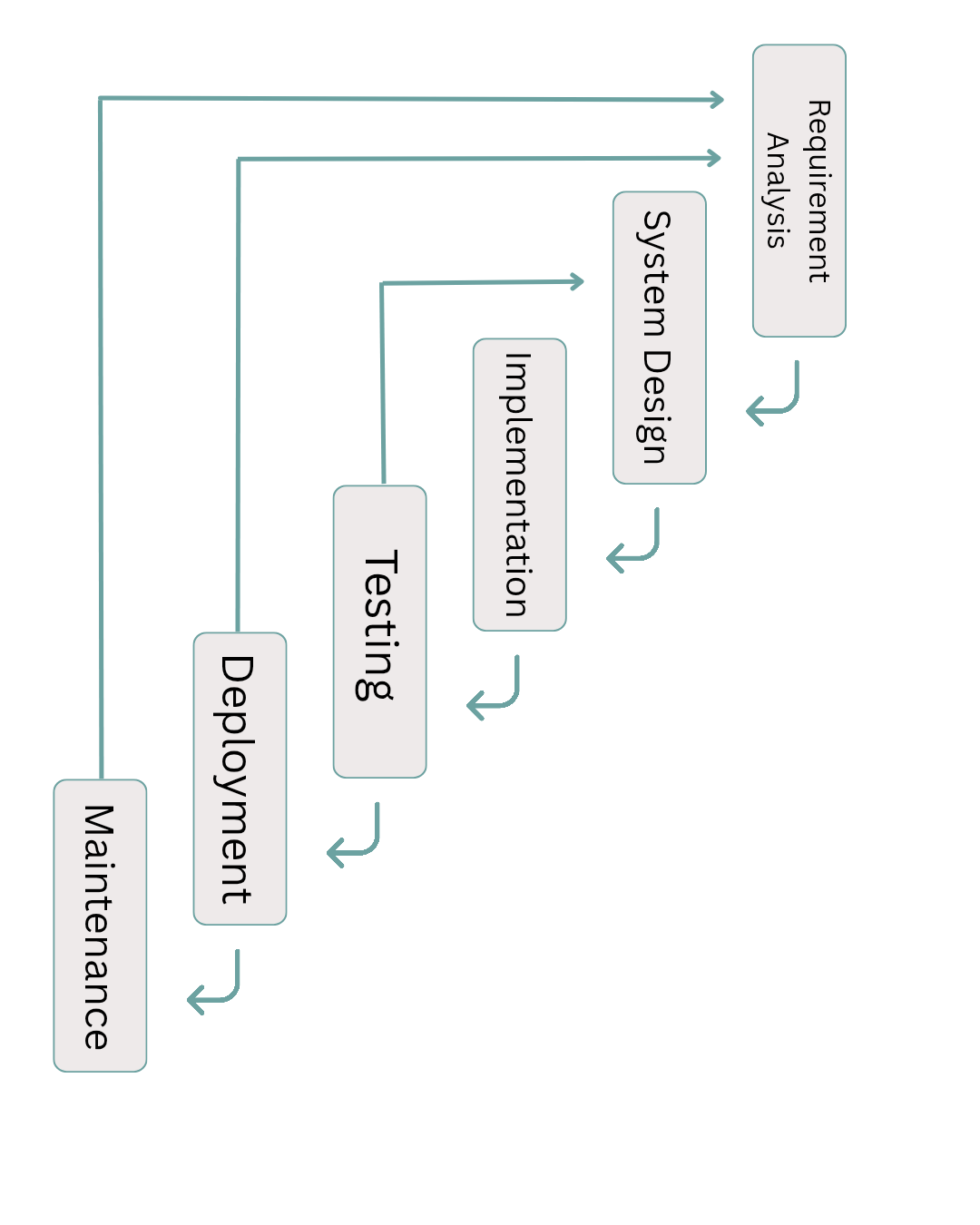
# Software Development Models and Procedures

The Waterfall Model introduced a structured and linear approach to software development. It begins with gathering requirements, followed by the design, implementation, testing, and deployment phases. In the Waterfall Model, each phase must be fully completed before the next phase can begin, and there is no overlap between stages. This strict sequential process ensures that each step is thoroughly addressed before moving forward. However, the lack of flexibility means that changes or adjustments after a phase is completed are difficult to accommodate, making it most effective for projects with clearly defined and stable requirements. (Senarath 2021).

The developer utilized this software development model, which encompasses a series of processes and methods, to achieve the desired outcome of the proposed system. They applied this model while considering factors such as a thorough understanding of system requirements, time constraints, addressing skill gaps and training needs as a software developer, and suitability for web application development.

# Figure 4

## Waterfall Model



Here are the tasks that a developer should undertake during the development of a system or website:

**Requirements Analysis.** The requirements analysis phase is a critical initial step in the development of the system. During this phase, the developer collaborates closely with stakeholders to examine and clarify project requirements. The primary objective is to gain a comprehensive understanding of the system’s goals, key functionalities, and any constraints related to the online application process. By gathering user needs, defining essential system features, and establishing clear project goals, the developer ensures alignment with client expectations and the overall project scope. Importantly, after the pre-oral defense, feedback from panelists may prompt a return to the requirements analysis phase. This iterative approach allows the development team to refine and enhance the initial findings based on expert recommendations, ensuring a more effective, user-centered, and tailored system.

**System Design.** In the system design phase, the detailed specifications are formulated based on the requirements gathered. This phase involves designing the overall system architecture, user interface layouts, database structures, and workflow diagrams. The aim is to create a clear and structured blueprint that will guide the development process. Careful consideration is given to the user experience, ensuring that the application process is intuitive, efficient, and accessible. The system design serves as a roadmap for developers, minimizing ambiguities and laying a solid foundation for successful implementation.

**Implementation**. The implementation phase involves translating the approved system design into a functional product. Developers write the actual code to build the online application system, integrating front-end and back-end components according to the design specifications. Key functionalities such as user registration, application submission, form validation, data storage, and notification systems are developed during this stage. Throughout the implementation, best coding practices are followed to ensure code quality, system security, and scalability. Close adherence to the design documents ensures that the developed system remains consistent with stakeholder expectations.

**Testing.** After the system is developed, thorough testing is conducted to identify and fix any defects, ensuring that the system performs as intended. Testing includes functionality testing, usability testing, security testing, and performance testing. Any bugs, inconsistencies, or areas for improvement are addressed promptly. The primary goal of this phase is to verify that the system meets all specified requirements, provides a smooth user experience, and can handle real-world scenarios reliably before it is officially deployed.

**Deployment** Upon Once the system passes the testing phase, it is deployed for live use. In this phase, the system becomes accessible to users, allowing them to register and submit their applications. Proper deployment planning ensures a smooth transition from development to production. During deployment, the team also monitors the system to quickly respond to any unexpected issues that may arise in the live environment.

**Maintenance**. Finally, the developer enters the maintenance phase, where they provide ongoing support and updates to the deployed system or website. This involves monitoring performance, addressing user feedback, and addressing any issues or bugs that may arise post-deployment. The developer also explores opportunities for enhancements and optimizations to improve the system's

functionality, reliability, and user experience over time.

# Software Cost Estimation

Software Cost Estimation (SCE) remains a complex and critical task within the field of software engineering, guiding project managers and developers in predicting costs at the early stages of the Software Development Life Cycle (SDLC). Accurate estimations of cost, time duration, and required effort are essential for the successful execution of software projects. Over the past decade (2011–2022), researchers have proposed various SCE models, including algorithmic, non-algorithmic, and learning-oriented approaches, aiming to enhance estimation accuracy. Comparative analyses based on methods, selected datasets, and evaluation metrics reveal that machine learning-based models generally outperform traditional techniques. Nevertheless, to the best of our knowledge and according to the existing literature, no universal model has yet emerged that can provide accurate estimations across all project types (Kumar, 2023).

The COCOMO model has led to innovative optimization techniques. Gandomani et al. (2022) proposed the integration of Genetic Algorithms (GA) with Environmental Adaptation (EA) methods to optimize the coefficients of the standard COCOMO model. Their findings revealed that the combined use of GA and EA not only addressed the divergence problems associated with GA but also achieved an 8% improvement in estimation accuracy compared to when the algorithms were applied separately.

Similarly, the exploration of machine learning techniques for software cost estimation has gained momentum. Kara and Şamli (2021) evaluated the performance of 27 machine learning algorithms within the Waikato Environment for Knowledge Analysis (WEKA), utilizing datasets such as COCOMO81, COCOMONASA, and COCOMONASA2. Their study highlighted that no single algorithm consistently outperformed others across different datasets; instead, algorithm performance varied depending on the dataset characteristics and parameter configurations. This underscores the complexity of selecting appropriate estimation models for diverse software project contexts.

Another notable advancement is the adaptation of existing estimation models to minimize the dependency on large historical datasets. Tahayori and Sami (2023) proposed a methodology that extracts CoBRA-required data directly from COCOMO datasets. By enabling organizations to use fewer historical data points while maintaining high estimation accuracy, this approach offers a practical solution to one of the major limitations of traditional COCOMO calibration processes. Application of this method to six public COCOMO datasets demonstrated improved estimation accuracy when compared to standard approaches.

The recent research has sought to enhance COCOMO-II by applying computational intelligence techniques. Ullah et al. (2021) introduced a biogeography-based optimization (BBO) method to refine the coefficients of the COCOMO-II model. Evaluations conducted on the NASA-93 and Turkish Industry software project datasets showed that the proposed BBO-COCOMO-II model achieved superior performance in terms of error minimization and estimation accuracy, outperforming several baseline models, including genetic algorithms, particle swarm optimization, and the flower pollination algorithm.

In summary, software cost estimation remains a foundational element in ensuring the success and sustainability of software projects. Continued enhancements to models like COCOMO through the integration of optimization algorithms, machine learning techniques, and hybrid data adaptation methods are crucial. These advancements not only improve estimation precision but also enable better project management practices, ultimately contributing to higher success rates in software development endeavors.

**REFERENCE**

Huang, J., Gao, M., Su, J. Chen, R. (2019). An application online signature method and system.

Ji, B., & Wang, S. (2022). Application of Signature Technology Based on RSA Algorithm in E-Commerce Security. 1–5. https://doi.org/10.1109/ICERECT56837.2022.10060107.

Feng, G., & Rongduan, C. (2019). Remote online electronic signature method based on WEB script data stream operation.

Lin, E., Zhang, P., Liu, X., & Li, J. (2019). An electronic signature method suitable for whole-process management.

Edisherashvili, T. (2020). Legal regalements of e-signature. The Journal of Legal Studies, 25(39), 98–127. https://doi.org/10.2478/JLES-2020-0006.

Sun, T. (2020). Electronic signature application method based on block chain.

Li, X., Zhong, Y., Huang, Y., Yuan, Y., & Xu, G. (2019). Electronic signature method and device of document, storage medium and electronic equipment.

A Comprehensive Study on Digital Signatures. (2023). International Journal of Advanced Research in Science, Communication and Technology, 37–39. https://doi.org/10.48175/ijarsct-11608.

Ohme, P. J., & Hellman, B. A. (2019). Electronic document signatures and signature retrieval involving mobile communication device.

Petcu, A., Frunzete, M., & Stoichescu, D. A. (2023). A Practical Implementation Of A Digital Document Signature System Using Blockchain. 1–6. <https://doi.org/10.1109/ATEE58038.2023.10108308>.

Rozi, N. F., Mauludin, M. N., & Rachman, A. (2022). Pengembangan Aplikasi Pelayanan Dokumen untuk Mahasiswa. Jurnal ELTIKOM : Jurnal Teknik Elektro, Teknologi Informasi Dan Komputer, 6(2), 174–185. <https://doi.org/10.31961/eltikom.v6i2.555>.

Supriya, Ch. G., Devi, T. B., Varsha, P., & Shaheed, S. (2024). Bus Pass Application and Renewal Using Cloud Computing. 144–148. https://doi.org/10.1109/icoeca62351.2024.00038

Nuraeni, A., & Hardianti, S. S. (2019). Aplikasi penerimaan karyawan online dengan fitur informasi jadwal tes dan hasil kelulusan. 2(1), 70–88. <https://doi.org/10.32627/INTERNAL.V2I1.66>.

Shi, J., Xu, X., Yang, Z. Y., Hou, B., Zong, X., & Sun, X. (2023). Construction and application of vehicle digital registration system. Third International Conference on Computer Vision and Data Mining (ICCVDM 2022), 12511, 1251109. <https://doi.org/10.1117/12.2660307>.

Li, Z. (2020). Online document filling and editing method and device and readable storage medium.

Rusli, Y. M. (2019). PENGARUH EFEKTIVITAS PENERAPAN e-FILING DAN MODERNISASI SISTEM PERPAJAKAN INDONESIA TERHADAP EFEKTIVITAS PEMROSESAN DATA PERPAJAKAN. 12(1). <https://doi.org/10.30813/JAB.V12I1.1509>.

Barrón-Adame, M., Rico-Moreno, L., Guzmán-Cabrera, R., García, E., & Quintanilla-Dominguez, J. (2019). Aplicación web para la administración de formatos y solicitud de comisión. 1–7. <https://doi.org/10.35429/JTI.2019.19.6.1.7>.

Shi, J., Xu, X., Yang, Z. Y., Hou, B., Zong, X., & Sun, X. (2023). Construction and application of vehicle digital registration system. Third International Conference on Computer Vision and Data Mining (ICCVDM 2022), 12511, 1251109. <https://doi.org/10.1117/12.2660307>.

Parmono, Z. A., Parjan, P., Andayani, T., Azzahra, V. N., & Akaike, H. (2024). A WEBSITE-BASED INFORMATION SYSTEM DEVELOPMENT FOR APPLYING TEMPORARY AIRPORT PASS REGISTRATION. Journal of Airport Engineering Technology (JAET), 5(1), 51-58.

Atianashie Miracle, A. (2023). The Application of Computer in Motor Vehicle Registration over Manual System. Int. J. Sci. Res. in Computer Science and Engineering Vol, 11(2).

Kußmaul, F., & Ruland, C. (2022). A secure hardware-based online vehicle registration system.

Nair, V., & Mohan, P. (Year). Student vehicle registration system with QR code. [Publisher or journal name, if applicable].

Oloyede, M. O., Oladele, O. T., Abdulrahaman, M. D., Sanni, S. A., Oyekola, M. O., Olabemiwo, D. O., & Muhammad-Awwal, A. B. (2024). Design of a Web-Based Vehicle Registration System Using QR Code. University of Ibadan Journal of Science and Logics in ICT Research, 11(1), 14-26.

Parmono, Z. A., Parjan, P., Andayani, T., Azzahra, V. N., & Akaike, H. (2024). A Website-Based Information System Development for Applying Temporary Airport Pass Registration. Journal of Airport Engineering Technology (JAET), 5(1), 51-58.

Dalisay, F. V. (2019). Development of a multi-platform online reservation system for car services. Asian Journal of Multidisciplinary Studies, 2(1), 165-173.

Turnip, F. F., & Turnip, A. (2020, June). Development of Online Ticket Booking Application for Ferry Crossing Website Based in Toba Lake Area. In 2020 3rd International Conference on Mechanical, Electronics, Computer, and Industrial Technology (MECnIT) (pp. 381-385). IEEE.

Anekar, S., Yeginwar, S., & Sonune, H. (2022). Automated gate system using number plate recognition (NPR). In ICT Systems and Sustainability: Proceedings of ICT4SD 2021, Volume 1 (pp. 413-420). Singapore: Springer Nature Singapore.

Mohamed, A. A., Shawky, M., Abdel-Latif, H. M., & Sabry, M. S. (2019). Enhancement of parking management system in Cairo using smartphones. International Journal of Engineering and Advanced Technology, 8(6), 3292–3300.

Priya, D. B., Rao, V. R., & Kumar, C. V. (2019). Intelligent parking system. International Journal of Recent Technology and Engineering (IJRTE), 8(2S3).

Maia (2019) identified quality characteristics specific to mobile applications by adapting the ISO/IEC 25010 quality standards.

Payongayong, R. T., & Palaoag, T. D. (2024). Evaluation of smart campus driveway and vehicle asset management system. Library of Progress – Library Science, Information Technology & Computer, 44(3).

Madanbhavi, L., Sawant, A. D., Bharamagoudar, G. R., Patil, M. R., & GS, H. (2021). Smart parking system for educational institution. Grenze International Journal of Engineering & Technology (GIJET), 7(1).

Kuang, Y., & Liu, Y. (2023). Analysis and design of the smart park entry logistics vehicle management system. Journal of Engineering, 1(2), 41.

Shaji, A., Joshy, A., Athulkrishna, M. J., Biju, J., & Thomas, B. (2023). College Bus Security and Management System.

Yitian, W. (2024). Visitor Management System with Decision Support. Journal of Educational Technology, 12(3).

Owoola, S. I. (2024). Mobile-based automated visitor management application (AVMA) for enhancing security in student housing.

Pabelona Jr, R. M., & Dormido, J. M. D. (2025). Keyless entry system using a smartphone for vehicle: A development of vehicle security performance. International Journal of Latest Technology in Engineering, Management & Applied Science, 14(1), 116-122.

Ghanem, S., Ali, S. T., Gouhary, K. G., Mahoud, M. M., Fshak, A. H., Shehata, M. M., & Soliman, H. H. (2022). Design and implementation of an integrated vehicle security system (IVSS). International Journal of Industry and Sustainable Development, 3(1), 87–97.

Liu, W., Yang, J., Song, Y., Yu, X., & Zhao, S. (2020, January). Research on software quality evaluation method based on process evaluation and test results. In 2019 6th International Conference on Dependable Systems and Their Applications (DSA) (pp. 480–483). IEEE. https://doi.org/10.1109/DSA.2019.00082

Promy, N., & Islam, S. (2019, February). A smart android based parking system to reduce the traffic congestion of Dhaka city. In 2019 21st International Conference on Advanced Communication Technology (ICACT) (pp. 124–128). IEEE. https://doi.org/10.23919/ICACT.2019.8701971

Gopal, D. G., Jerlin, M. A., & Abirami, M. (2019). A smart parking system using IoT. World Review of Entrepreneurship, Management and Sustainable Development, 15(3), 335-345.

Lei, Y., & Qi, J. (2019). A method of vehicle dashboard image location based on machine vision. International Conference on Information Science and Control Engineering. https://doi.org/10.1109/ICISCE48695.2019.00024

Kibwana, M. O., & Ngugi, M. W. (2023). Digitalization of Vehicle Road Documents System using Innovative Technology: A Case Study of the Circulation Road Police - DRC. International Journal of Technological Innovations, 12(1), 45–58.

Maduka, N. C., Ozuomba, S., & Ekott, E. E. (2020). Internet of Things-Based Revenue Collection System for Tricycle Vehicle Operators. In Proceedings of the 2020 International Conference in Mathematics, Computer Engineering and Information Technology (MCEIT). IEEE. https://doi.org/10.1109/MCEIT49285.2020.9240992

Mary, L., & Koshy, B. I. (2023). Detection and classification of vehicles using audio visual cues. Multimedia Tools and Applications, 82(28), 44087-44106.

Grimm, D., Stang, M., & Sax, E. (2021). Context-aware security for vehicles and fleets: A survey. IEEE Access, 9, 101809-101846. https://doi.org/10.1109/ACCESS.2021.3074556

Nasr, T. (2021). Large-scale study of internet-connected electric vehicle charging station management systems: Discovery, security analysis and mitigation (Doctoral dissertation, Concordia University). https://spectrum.library.concordia.ca/id/eprint/988799/

Ogbe, B. (2018). Design and implementation of a web-based Vehicle Licensing System (Doctoral dissertation, Godfrey Okoye University).

Sheddi, R., & Umarani, M. V. (2017). E-verification of driving license through aadhaar database. INTERNATIONAL JOURNAL OF ENGINEERING DEVELOPMENT AND RESEARCH, 5(3), 347-350.

Sharma, J., Sindhu, G. S., Sejwal, S., Solanki, J., & Majumdar, R. (2019, February). Intelligent vehicle registration certificate. In 2019 Amity International Conference on Artificial Intelligence (AICAI) (pp. 418-423). IEEE.

Thammarak, K., Sirisathitkul, Y., Kongkla, P., & Intakosum, S. (2022). Automated data digitization system for vehicle registration certificates using Google Cloud Vision API. Civil Engineering Journal, 8(7), 1447-1458.

Stephen, M. (2020). Vehicle-data analytics. https://www.freepatentsonline.com/y2020/0017117.html.

Lin, Q., Zhu, R., Chen, Y., & Chen, H. (2019). On-vehicle analysis system based on sign big data.

Hironobu Sugimoto, et al. Vehicle Data Analysis Method and Vehicle Data Analysis System. 15 May 2012, https://patents.google.com/patent/CN103562978B/en.

Ahmed, T., & Kim, H. (2019). Predictive analytics for streamlining vehicle registration processes. Journal of Transport Technology and Innovation, 6(2), 112–121.

Li, X., Wang, Y., & Zhao, L. (2020). Smart permit systems for urban traffic management: A data-centric approach. Transportation Science Review, 13(1), 75–88.

Gupta, P., & Sharma, R. (2021). Machine learning-based permit approval system for transport authorities. International Journal of Intelligent Systems, 9(3), 144–153.

Kumar, R., & Devi, M. (2020). Role of IoT and data analytics in vehicle monitoring and control systems. International Journal of Advanced Transportation Systems, 11(2), 78–88.

Chandra, S. (2021). Optimizing vehicle permit systems with cloud-based data analytics. Cloud Computing and Transportation Management, 8(1), 45–56.

World Bank. (2019). Data-driven public services: Innovations in government permit systems. World Bank Publications.

Warner, R. C. (1975). Analysis of motor vehicle ownership and operation in higher education. Indiana University. https://www.usi.edu/public-safety/parking/vehicle-registration.

Palomino, B. J., Peñafiel, C. L. S., & Yala, B. C. (2015). De La Salle University-Dasmariñas College of Liberal Arts and Communication Social Sciences Department Bachelor of Arts in Political Science. Loss\_of\_Productivity\_as\_an\_Effect\_of\_Traffic\_Congestion-libre.pdf.

Sunio, V., Schmöcker, J. D., & Estuar, M. R. J. (2017, October). Implementation of a mobility behavior change support system in Manila Philippines. In *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)* (pp. 661-666). IEEE. Implementation of a mobility behavior change support system in Manila Philippines | IEEE Conference Publication | IEEE Xplore.