Journal of Data Analytics and Intelligent Decision-making Vol. 2 Issue. 1 (2026) 58-79

Journal of Data Analytics and Intelligent Decision-Making

Journal homepage: jdaid.qom.ac.ir

University of Qom

Data-Driven Discrete-Event Simulation of Open Banking API Access Granting

Leila Mahdavi [a]*; Mohammad Reza Fathi [b]; Zahra Ansarifard-Joshan[a]

a. Department of Industrial Engineering, Faculty of Engineering, College of Farabi, University of Tehran, Tehran, Iran.
b. Associate Professor, College of Farabi, University of Tehran, Iran.

ARTICLEINFO

Keywords:
Data-driven discrete-eventsimulation
Open banking
Event log analysis
Staffing optimization
Workforce allocation

ABSTRACT

Open Banking platforms must manage volatile application programming interfaces (APIs) request streams alongside complex, multi-step approval workflows. This study investigates operational bottlenecks and workforce optimization within a multi-stage request review process, with direct implications for waiting time, turnaround time, and overall system responsiveness. A dynamic representation of the workflow is developed using data-driven discrete-event simulation in the AnyLogic environment. Unlike purely theoretical models, the proposed framework incorporates real operational records: inter-arrival times are derived from historical logs and embedded into the simulation to reproduce realistic traffic fluctuations. Each incoming request is modeled as an agent characterized by specific attributes and routed through a queueing network consisting of preliminary assessment, specialist evaluation, and technical validation stages. Experimental results demonstrate that integrating actual event sequences significantly enhances model fidelity, particularly in capturing peak-load dynamics and stress conditions. The model enables scenario-based analysis of staffing configurations and resource allocation strategies. Ultimately, the study delivers a decision-support tool that helps managers balance employee utilization with service quality while improving process stability and operational performance in banking ecosystems.

Introduction

The financial-services industry has undergone a rapid shift from closed, institution-centric architectures toward open, interoperable digital ecosystems in which banks, fintechs, and third-party providers co-create customer value. An aspect of this transformation is Open

* Corresponding author.
E-mail addresses: l.mahdavi@ut.ac.ir (L. Mahdavi), reza.fathi@ut.ac.ir (M.R. Fathi), zahraansarifard.18765a@gmail.com (Z.A. Joshan)
Received 12 Jan 2026; Received in revised form 23 Feb 2026; Accepted 26 Feb 2026
Available online 30 Mar 2026

Cite this article: Joshan, A. (2026). Data-Driven Discrete-Event Simulation of Open Banking API Access Granting. *Journal of Data Analytics and Intelligent Decision-Making*, 2(1), 58-79.
https://doi.org/10.22091/10.22091/jdaid.2026.15365.1033

Banking, typically defined as a customer-consented data and service-sharing regime enabled by standardized application programming interfaces (APIs). Recent bibliometric and systematic syntheses emphasize that Open Banking is not merely a technical interface upgrade; it constitutes an ecosystem paradigm that reshapes competition, innovation, and governance through consent-based portability of financial data and the reconfiguration of industry roles (Barbosa Casolaro et al., 2025; Briones de Araluze & Cassinello Plaza, 2022). Complementing this view, evidence from the United States indicates that voluntary adoption of external APIs by banks has increased substantially over time and is associated with measurable changes in bank performance and information flows underscoring that APIs are becoming strategic assets rather than peripheral IT components (Lin et al., 2025). At the core of Open Banking, APIs operationalize controlled access to services such as account information, payment initiation, identity verification, and transaction management. When provided through developer portals and sandboxes, these APIs enable third parties to design and test new products, shorten time-to-market, and scale digital financial services. More broadly, fintech research suggests that such platformization is central to the ongoing "fintech revolution," in which new digital capabilities and business models are re-organizing financial operations and customer experience (Gomber et al., 2018). At the same time, regulatory and cybersecurity requirements particularly in jurisdictions influenced by PSD2 introduce additional constraints on API exposure, authentication, and secure communication. For example, analyses of PSD2 compliance highlight that mandated third-party API access can improve consumer choice and competition but also expands the attack surface, raising privacy and security risks and creating implementation challenges for incumbent institutions (Gounari et al., 2024). Security-focused studies on Open Banking standards further show that the protocols governing authorization and data exchange must be carefully specified and verified to ensure the confidentiality and integrity of sensitive financial information (Modesti et al., 2025). While the strategic and regulatory significance of Open Banking is increasingly well documented, operational execution remains a major determinant of whether Open Banking platforms deliver reliable and scalable service. In practice, a bank's Open Banking capability hinges on its end-to-end workflow for onboarding third parties and granting API access. These workflows are typically multi-stage and resource-constrained, involving initial screening, expert review (e.g., risk, compliance, and technical assessment), and final testing/activation in sandbox or production environments. As a result, they behave as queueing systems with stochastic arrivals, heterogeneous request types, variable service times, and potential rework loops caused by incomplete documentation or failed tests.

In Open Banking ecosystems where demand can be volatile, inefficient management of the API access-granting pipeline can directly degrade key performance indicators (KPIs) such as responsiveness, waiting time, and overall throughput, ultimately limiting ecosystem participation and slowing innovation. From a decision-support perspective, these characteristics make the API granting process difficult to evaluate using static capacity calculations or deterministic process maps. Instead, methods are needed that can capture dynamic congestion effects, priority rules, routing decisions, and the interaction of multiple resource pools under uncertainty. Discrete-event simulation (DES) is a well-established approach for analyzing such systems as it represents process evolution as a sequence of events, explicitly models randomness, and enables "what-if" experimentation without disrupting live operations. DES has been widely used to identify bottlenecks and evaluate capacity and staffing policies in complex service settings, especially where demand and service times exhibit significant variability (Robinson, 2005). A further challenge is ensuring that simulation inputs reflect real demand dynamics. Event-log data generated by information systems can provide empirically grounded arrival patterns and activity timings, and the process-mining literature has shown how

event data can be used to discover, validate, and improve process models (Van Der Aalst, 2012). Building on this, research has demonstrated that simulation models can be discovered or calibrated using event logs, thereby increasing model fidelity and supporting more credible performance analysis (Rozinat et al., 2009). Integrating event-log evidence into simulation is particularly relevant for Open Banking workflows, where bursty arrivals or policy-driven surges (e.g., ecosystem growth initiatives) can change congestion patterns and expose hidden capacity risks.

Motivated by this gap between ecosystem-level narratives and operational execution, this study develops a data-driven discrete-event simulation model of a multi-stage API access-granting process for Open Banking platforms. The model represents each API request as an individual entity with attributes, such as request type and routing outcomes, moving through an explicit queueing network that includes initial approval, expert evaluation, and technical testing/activation stages. By injecting inter-arrival times derived from historical records and by capturing stage-dependent service-time distributions, the model enables systematic bottleneck diagnosis and scenario analysis of human-resource allocation. The contribution is twofold: (i) empirically grounded modeling of the API access-granting pipeline as a stochastic service system, and (ii) actionable insights into how staffing and policy choices influence waiting time, throughput, and service quality in Open Banking operations. Ultimately, the proposed approach supports managers in balancing resource utilization with responsiveness, helping Open Banking platforms scale securely and reliably as ecosystem participation grows.

Literature Review

This literature review is organized in two parts. Part A summarizes DES and recent data-driven simulation approaches that integrate event logs and process mining for modeling multi-stage queueing workflows. Part B reviews Open Banking as an API-enabled ecosystem, covering governance, security, and the operational implications of multi-party access and approval processes.

Part A. Discrete-Event Simulation and Data-Driven Process Simulation

DES models system evolution as a sequence of discrete events, making it well suited for stochastic queueing networks and resource-constrained workflows such as multi-stage request review processes. Foundational simulation methods formalize conceptual modeling, input analysis, verification/validation, and experimentation to estimate waiting time, throughput, and utilization under uncertainty (Banks et al., 2013; Law, 2014; Robinson, 2005).

A key limitation of purely theoretical DES is input realism. Process mining provides an empirical basis by extracting process structure and timing from event logs, supporting evidence-based model calibration. Prior studies show that simulation models can be discovered or aligned to event data, improving fidelity for bottleneck diagnosis and what-if scenario analysis (Rozinat et al., 2009; Van Der Aalst, 2012).

Recent research advances this integration through automated discovery of business-process simulation models from event logs and hybrid approaches that combine process discovery with deep learning for more accurate routing and timing behavior. Work on probabilistic resource calendars and simulation-based predictive process mining further extends data-driven DES toward realistic capacity constraints and forward-looking performance assessment (Bocciarelli & D'Ambrogio, 2024; Camargo et al., 2020, 2022; López-Pintado & Dumas, 2023).

Since multi-stage approval processes often exhibit shifting constraints, DES is frequently combined with explicit bottleneck detection methods. Studies that join DES with process mining for hierarchical bottleneck identification are directly relevant to diagnosing congestion in API access granting pipeline (Septiyanto et al., 2022).

Finally, scalability and control-oriented simulation are emerging topics. Fast approximations for DES of queueing networks and differentiable DES approaches support more efficient experimentation and policy search, providing foundations for simulation-optimization of staffing and resource allocation in digital service systems (Che et al., 2024; Wang et al., 2023).

Part B. Open Banking (APIs, Governance, and Cybersecurity)

Open Banking is typically defined as customer-consented sharing of financial data and services through standardized APIs. Systematic and bibliometric syntheses describe it as an ecosystem-level innovation that reshapes competition, platform governance, and stakeholder roles. More broadly, fintech research positions APIs and platformization as central mechanisms of digital transformation in financial services (Briones de Araluze & Cassinello Plaza, 2022; Casolaro et al., 2025; Gomber et al., 2018).

Evidence suggests that banks' external API adoption has increased and may be associated with changes in information flows and performance, supporting the view that APIs are strategic assets. Industry-oriented scholarship also emphasizes the 'API economy' and embedded-finance dynamics, where standardized interfaces enable rapid product composition, onboarding through portals/sandboxes, and ecosystem scaling (Gounari et al., 2024; Lin et al., 2025; Modesti et al., 2025).

Regulatory and security requirements complicate this expansion. PSD2 compliance analyses highlight that mandated third-party API access can improve consumer choice but expands the attack surface, creating implementation challenges and higher security demands. Formal protocol analyses indicate that authorization and data-exchange mechanisms must be precisely specified and verified to preserve confidentiality and integrity of sensitive financial information (Adeleke et al., 2024; Ojehomon et al., 2026; Veldurthi, 2025).

Technology-focused studies address trust foundations, such as identity, consent, and access control. Blockchain-based proposals—including BIMAC and key-revocation access control—aim to strengthen decentralized consent management, authentication, monitoring, and revocation capabilities. Complementary work highlights layered security architectures (e.g., MFA, OAuth-based authorization, TLS, and API gateways) as well as cyber-risk governance for API-enabled financial crime (Hota, 2022; Liao et al., 2022; Shacheendran et al., 2025; Sharma, 2025).

AI and data-driven innovation are increasingly linked to Open Banking. Sharma (2025) emphasized tensions between AI's data intensity and strict financial-data regulation, proposing governance and API-mediated approaches for integrating AI while maintaining confidentiality and compliance. Broader Open Banking framework discussions also emphasize interoperability, standardization, and user-centric architecture as prerequisites for scalable innovation (Mohammed, 2024; Pandey et al., 2025).

Empirical studies indicate that Open Banking can influence market entry and consumer outcomes. Cross-country evidence links Open Banking to higher fintech entry and investment, while other global analyses associate Open Banking initiatives with higher likelihood of formal saving and digital remittance, supporting claims of financial inclusion impacts under supportive policy and market conditions (Hossain, 2025; Muqorobin et al., 2021).

Implementation research shows that adoption trajectories depend on legal alignment and ecosystem coordination. Comparative legal work (e.g., Ukraine vs. EU) and country readiness studies (e.g., Indonesia) underscore the importance of governance capacity and operational capability; trend analyses further document the fast-growing and diversifying research landscape (Babina et al., 2025; Niankara et al., 2025; Riad & Elhoseny, 2022).

Sharma (2025) explored the integration of Artificial Intelligence (AI) with Open Banking and highlighted the challenges arising from the conflict between AI's data requirements and strict financial data regulations. The study proposed a structured framework that embeds AI into Open Banking ecosystems through strong data governance and API-based model integration, ensuring data security while leveraging AI capabilities. Liao et al. (2022) proposed a blockchain-based identity management and access control framework for Open Banking ecosystems to enhance digital identity integration and privacy-preserving data sharing. The framework, named BIMAC, leverages smart contracts and stateless authentication to enable secure user consent management, decentralized authentication, and monitoring of third-party service provider access. Performance evaluation showed that the proposed solution is compatible with existing systems and effectively improves security and trust in Open Banking environments. Mohammed (2024) investigated the role of Open Banking frameworks and APIs in reshaping the financial ecosystem. The study analyzed regulatory and technological drivers enabling secure data sharing between banks and third-party providers, highlighting improvements in innovation, customer satisfaction, financial inclusion, and personalized financial services. The findings also emphasized challenges related to data security, privacy risks, and the need for standardized API protocols to support a decentralized, user-centric financial system. Pandey et al. (2025) examined cybersecurity challenges in Open Banking environments, with a focus on securing APIs and protecting customer data. The study evaluated the effectiveness of security mechanisms such as multi-factor authentication, OAuth 2.0, Transport Layer Security, and API gateways through case studies and compliance analysis. The findings emphasized the necessity of multi-layer security architectures to mitigate data leakage and unauthorized access in Open Banking ecosystems. Muqorobin et al. (2021) investigated the impact of Open Banking based on Open APIs on the sustainability of the banking sector in Indonesia. Using a qualitative phenomenological approach, the study assessed the readiness of Indonesian banks to adopt Open Banking by analyzing international experiences, particularly the United Kingdom. The findings indicated that Open API–based Open Banking is feasible in Indonesia and can enhance digital transformation and connectivity between banks and payment system service providers.

Hossain (2025) analyzed the cybersecurity implications of Open Banking and APIs in the financial sector. Using case studies, vulnerability assessments, and expert interviews, the study identified key security risks such as data breaches and third-party threats. The findings emphasized the adoption of robust mitigation strategies, including secure API development, strong authentication mechanisms, regulatory compliance, and Zero Trust–based architectures, to protect sensitive financial data while supporting innovation in Open Banking ecosystems. Babina et al. (2025) provided early empirical evidence on the effects of Open Banking–enabled customer data access on fintech market entry. Using a novel cross-country dataset covering 49 adopting countries, the study showed that Open Banking policies lead to a 50% increase in fintech venture capital investment, with stronger effects under more comprehensive regulatory frameworks. However, a quantitative model revealed trade-offs, indicating that while customer-directed data sharing enhances market entry and product offerings, it may negatively affect some consumers and reduce incentives for ex-ante information production. Riad et al. (2022) proposed a blockchain-based key-revocation access control scheme for Open Banking to protect cloud-hosted financial data from unauthorized access. The approach, termed BKR-AC, utilizes smart contracts deployed on the Ethereum platform to enable dynamic key revocation even after customer authentication. Experimental evaluations demonstrated that the proposed scheme offers faster response time than traditional certificate revocation methods (CRL and OCSP) for non-revoked keys, acceptable communication overhead, and robust security against common Open Banking attacks. Niankara et al. (2025) investigated the global impact of Open Banking on consumer saving and digital remittance behaviors using a mixed-methods approach

combining bibliometric analysis and geospatial econometric modeling across 139 countries. The results indicated that Open Banking initiatives significantly increase consumers' likelihood of formal saving and digital remittance, with stronger effects observed in PSD2-regulated and market-driven environments. The study also highlighted the need for greater international research collaboration and provided empirical support for Open Banking as a driver of financial inclusion and sustainable development goals.

Methods

This study adopts a DES approach combined perspective to analyze the performance of the API access granting process within an Open Banking ecosystem. The primary objective is to identify operational bottlenecks arising from limited resources and stochastic arrival/service dynamics, and to evaluate workforce allocation scenarios aimed at improving KPIs. The simulation problem was first formulated by defining the system boundary: the process starts when a client/request enters the system and covers initial registration, request submission and document delivery, waiting in queues, completion of review activities, and final API delivery. Once the API is granted, the corresponding agent leaves the system and the simulation for that entity terminates. The model components entities, inputs, and stochastic parameters were then specified and the process was represented as a multi-stage queueing network. In the proposed model, each API request is modeled as an agent with dedicated attributes that traverses three service stations: initial assessment, expert review, and API technical testing. Resource constraints (reviewers/testers) combined with uncertain inter-arrival and service times lead to queue formation at different stages, affecting waiting time, response time, and overall system responsiveness; therefore, these measures are used as the core KPIs for system evaluation.

To make the model data-driven and improve the fidelity of traffic pattern reconstruction, real data were used both to derive the arrival pattern and to estimate acceptance/rejection behavior. The required inputs for parameterization include inter-arrival time between consecutive clients/requests, service times for initial assessment, expert review, and technical testing (all dependent on API type), and approval/rejection rates in the decision stages. For this purpose, the BPI Challenge 2017 dataset was selected. To extract inter-arrival times, the raw event log was processed by scanning the XES file using Python with the Pandas and PM4Py libraries in Visual Studio Code. Events corresponding to the start of a case (activity name "Create Application") were filtered, and the resulting timestamps were exported as a CSV file. Subsequently, the CSV data were used in Excel to compute inter-arrival times by differencing successive arrival timestamps; this enabled analysis of arrival distributions across time periods and injection of the derived arrival sequence into the simulation. Because stochastic inputs and process behavior depend substantially on API type and because API categories do not occur with the same frequency, API types were grouped prior to modeling to better define and analyze type-dependent variables. Based on available information, APIs were classified into five categories. The average demand frequencies over a 25-month period were used to generate the request mix in the simulation, as summarized in Table 1.

Table 1
API Categories and Estimated Demand Frequency

No.	API Category	Estimated Demand	Cumulative Demand
1	Core Banking APIs	34	34
2	Payment Execution and Consumer Services APIs	28	62
3	Digital Identity, Trust, and Compliance APIs	18	80
4	Extended Financial Ecosystem APIs	14	94
5	Operational Infrastructure APIs	6	100

Next, acceptance/rejection logic was incorporated as a function of API type. The approval probability after initial assessment was set according to Table 2.

Table 2
Approval Probability after Initial Assessment by API Type

No.	API Type	Approval Probability
1	Core Banking APIs	0.85
2	Payment Execution and Consumer Services APIs	0.78
3	Digital Identity, Trust, and Compliance APIs	0.65
4	Extended Financial Ecosystem APIs	0.50
5	Operational Infrastructure APIs	0.30

To estimate acceptance/rejection in the expert review stage, manual analysis of all 31,509 cases was not feasible due to data volume. Therefore, a script was developed to count final events directly in memory. The event O_Accepted was treated as Granted, while O_Refused and O_Cancelled were treated as Rejected/Cancelled. Based on the extracted counts, the grant rate was set to 40.23% and the reject/cancel rate to 59.77%. Service times were defined as type-dependent to capture operational variability and uncertainty. For each API category, a Uniform (a, b) distribution was assigned for each stage. Initial assessment service times are presented in Table 3, expert review service times in Table 4, and technical testing service times in Table 5.

Table 3
Initial Assessment Service Time by API Type (Minutes)

No.	API Type	Initial Assessment Time
1	Core Banking APIs	Uniform (20, 35)
2	Payment Execution and Consumer Services APIs	Uniform (30, 45)
3	Digital Identity, Trust, and Compliance APIs	Uniform (40, 60)
4	Extended Financial Ecosystem APIs	Uniform (55, 75)
5	Operational Infrastructure APIs	Uniform (70, 95)

Table 4
Expert Review Service Time by API Type (Minutes)

No.	API Type	Expert Review Time
1	Core Banking APIs	Uniform (45, 80)
2	Payment Execution and Consumer Services APIs	Uniform (60, 100)
3	Digital Identity, Trust, and Compliance APIs	Uniform (75, 120)
4	Extended Financial Ecosystem APIs	Uniform (90, 150)
5	Operational Infrastructure APIs	Uniform (120, 180)

Table 5
Technical Testing Service Time by API Type (Minutes)

No.	API Type	Technical Testing Time
1	Core Banking APIs	Uniform (15, 30)
2	Payment Execution and Consumer Services APIs	Uniform (25, 45)
3	Digital Identity, Trust, and Compliance APIs	Uniform (35, 60)
4	Extended Financial Ecosystem APIs	Uniform (50, 85)
5	Operational Infrastructure APIs	Uniform (70, 120)

After parameterization, the model was implemented and executed in AnyLogic 8.9.7 Professional. To support adaptive behavior, the client/request agent was customized and

assigned parameters related to API type, stage-specific service times, pass/fail probabilities, and routing logic across the queueing network. In the analysis phase, the study focused on extracting KPIs, including mean waiting time per stage, bottleneck identification, and the empirical distribution of delays, using statistical plots to assess not only average performance but also variability and tail behavior. Finally, multiple workforce allocation scenarios were simulated by varying staffing levels in the initial assessment, expert review, and technical testing stages; KPI comparisons across scenarios enabled decision-making aimed at achieving an optimal balance between workforce utilization and service quality.

Simulation Modeling and Problem-Solving Framework

This study develops a data-informed Discrete-Event Simulation model to analyze and improve the API access granting workflow in an Open Banking–fintech platform. The system is defined as the end-to-end process of accepting and granting API access to clients. The entry point is the moment a client submits a request for a specific API service (e.g., a balance inquiry API), and the exit point is either (i) successful issuance of the final access credential (API key) after completing all review and testing steps, or (ii) rejection due to technical and/or legal non-compliance. The system boundary covers internal activities including document verification, expert evaluation, and the technical testing stage, while exogenous factors such as regulatory changes and telecommunication infrastructure outages are treated as external and are not explicitly modeled.

The simulation is specified through standard DES components. Entities/agents represent incoming client requests; each associated with an API type. Resources are limited-capacity human staff responsible for processing and testing requests, grouped into three pools: InitialStaff (initial assessment), ExpertStaff (expert review), and TestStaff (technical testing). The system state is tracked via operational counters and queue-related measures (e.g., number of requests in each queue, total rejected requests, total granted requests). Core events include request arrival, completion of each service stage, exit after rejection, and exit after successful granting. Process dynamics are driven by stochastic inputs: the arrival stream of requests (inter-arrival times), stage-specific service times, and acceptance/rejection outcomes. These inputs are modeled as API-type dependent to reflect unequal demand frequency and differing risk/effort profiles across API categories.

The workflow is implemented as a multi-stage queueing network in AnyLogic Software. Each request follows a sequential path through three main stages of initial assessment, expert review, and API technical testing, and may experience waiting in three corresponding queues: the initial approval queue, the expert/technical review queue, and the final approval/issuance queue. Processing at each stage is modeled with standard AnyLogic blocks (Source–Queue–Service–Sink). Arrivals are generated via a Source block parameterized by inter-arrival times. Waiting is modeled using Queue blocks, and processing is modeled using Service blocks whose service-time distributions are selected conditionally based on the agent's API type. After the initial assessment, two outcomes are modeled: approved requests proceed forward, while requests with documentation issues are routed back to the start of the initial queue to represent rework/resubmission. After expert review, rejected requests exit the system, whereas accepted requests proceed to the technical testing stage and, upon completion, exit as successfully granted cases.

To represent request heterogeneity and enable per-type reporting, each request is modeled as a customized Client Agent characterized by four key attributes: API Type (categorical, controlling conditional service logic and routing), arrival Time (timestamp used for cycle-time and delay computation), is Approved (Boolean outcome of initial assessment), and is Expert Approved (Boolean final outcome after expert review). Additional model variables maintain

operational counts, including total arrivals (totalClient), number rejected (v_rejectedCount), and the number of entities in each queue (N_qInitialReview, N_qExpertReview, N_qTestAPI). Staffing levels are exposed as tunable parameters (num_Initialstaff, num_Expertstaff, num_Teststaff) and linked to the capacity of the corresponding resource pools to support scenario experimentation.

Performance evaluation focuses on queueing and service KPIs that reflect both client experience and operational efficiency. The primary indicators include: mean waiting time in the initial approval queue, mean waiting time in the expert/technical review queue, queue lengths in the corresponding stages, total number of non-approved clients, and total time in system (cycle time). These KPIs are used to identify bottlenecks and quantify the effect of staffing changes. Output analysis is conducted using built-in statistical and visualization elements (e.g., time plots for queue size and resource utilization, histograms for cycle-time distributions, and summary charts for granted vs. rejected proportions), enabling both aggregate and API-type-stratified comparisons. To manage input uncertainty and enable repeated experiments, empirical inter-arrival data extracted from the event log are fitted to a theoretical distribution using Minitab Software; candidate fits are screened using a p-value threshold ($p > 0.005$), and the best acceptable fit is selected based on the lowest Anderson–Darling statistic, resulting in a Lognormal model for inter-arrival times. The fitted distribution is then used to generate arrival samples for simulation runs. Overall, the problem-solving strategy treats the API granting workflow as an event-driven system and uses simulation as a decision-support "virtual laboratory" to balance operational cost (staffing levels across the three resource groups) against service performance (waiting times, responsiveness, and grant outcomes). By explicitly modeling limited resources, API-type-dependent variability, and stochastic arrivals, the framework enables robust bottleneck identification and evidence-based workforce allocation before implementing operational changes in the real system.

Results

The simulation model was executed in AnyLogic for 1000 hours (model time unit: minutes). Baseline results were collected using the model's built-in analytical and statistical components.

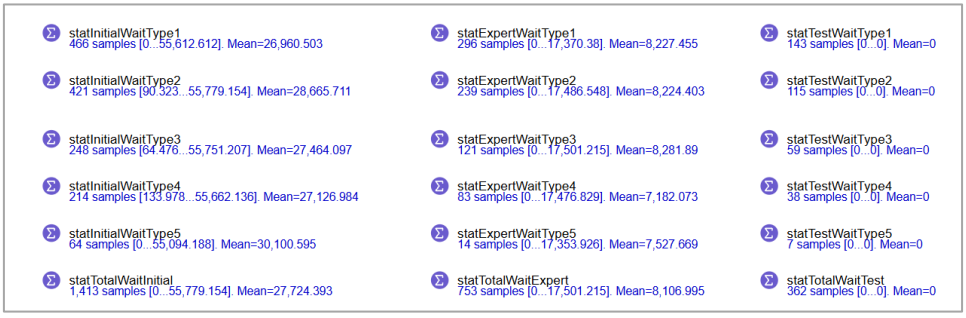

Figure 1
Simulation Experiment Results – Statistics Elements Output

Stage-level performance indicators, particularly queue size and mean waiting time, were recorded via the Statistics elements. These outputs are reported both by API type and in aggregated form for each queue (Figure 1).

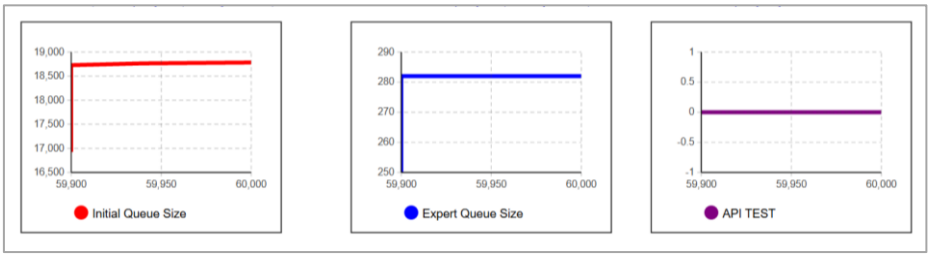

Figure 2
Instantaneous Queue Length Plots of the Model (Timeplot)

To observe transient behavior during the run, time-series plots were used to track the instantaneous queue lengths in the initial review, expert review, and testing stages over time (Figure 2).

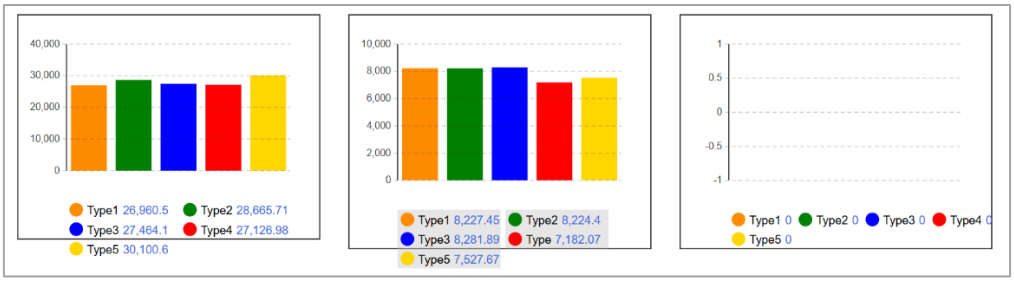

Figure 3
Queue Sizes by API Type for Initial Review, Expert Review, and Testing (Bar Chart)

Queue accumulation patterns were further examined using bar charts that compare queue sizes across API categories for each stage (Figure 3).

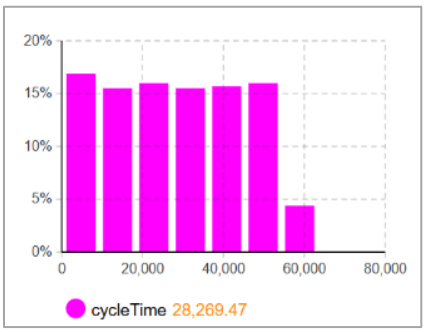

Figure 4
Histogram of Clients' Total Time in System (Cycle Time Distribution)

Overall client experience was assessed through the distribution of total time in system (cycle time). A histogram was generated to visualize the spread and variability of cycle times (Figure 4).

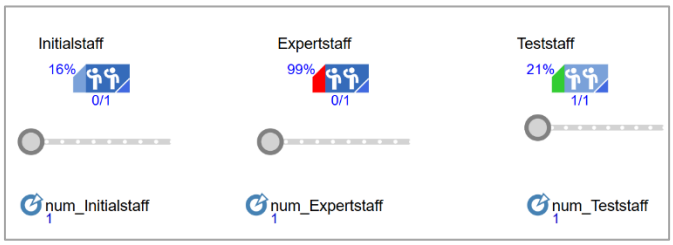

Figure 5
Staff Utilization Rates by Resource Group (Initial, Expert, Test)

Workforce performance was evaluated through utilization metrics. The utilization of the three resource pools (InitialStaff, ExpertStaff, and TestStaff) is reported as summary percentages (Figure 5) and as a graphical utilization profile (Figure 6).

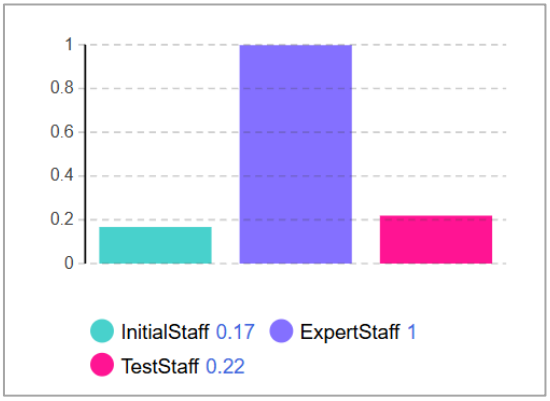

Figure 6
Graphical Comparison of Staff Utilization Over Time

Figure 7
Counts of Rejected Vs. Granted Clients (Dashboard Text Output)

Service outcomes were captured by the dashboard indicators that display the counts of rejected versus granted requests during the baseline run (Figure 7). The same information is also presented as a proportion-based visualization using a pie chart (Figure 8).

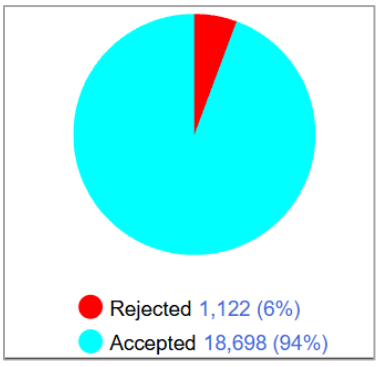

Figure 8
Rejected Vs. Granted Clients

Finally, additional dashboard indicators and tracked variables were recorded to support cross-checking of baseline system behavior and to provide a consolidated snapshot of key run statistics (Figure 9).

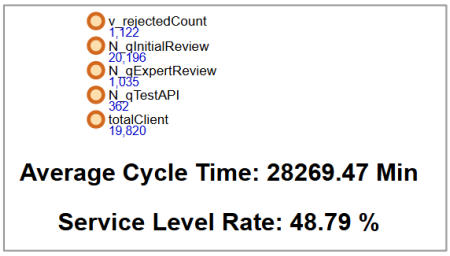

Figure 9
Additional Key Outputs From the Simulation Experiment (Dashboard Indicators)

The baseline dashboard indicates severe congestion and instability. The Average Cycle Time is approximately 28,269 minutes, meaning that a typical request remains in the system for about 19 days on average (Figure 4). The Service Level Rate is 48.79% (Figure 9), implying that fewer than half of incoming requests are served under the desired performance condition used in the model. At the same time, the model reports that 94% of requests are ultimately accepted (Accepted) (Figures 7 and 8). This combination suggests that while most requests eventually pass the approval logic, the operational value of "acceptance" is diminished by excessive waiting times and prolonged end-to-end delays. A key pattern in the baseline results is an apparent contradiction between a long initial review queue and low utilization of InitialStaff (16%) (Figures 2, 5, and 6). In queueing systems, a long upstream queue coupled with low upstream utilization typically signals downstream blocking rather than insufficient demand. Two factors explain why q_InitialReview remains large while *InitialStaff* is idle for significant periods. First, the next stage is saturated: ExpertStaff utilization is approximately 99%, indicating that expert review is operating at (or near) full capacity and cannot accept additional work (Figures 5 and 6). Consequently, after completing an initial review, an upstream server may be unable to transfer the case forward because the expert stage cannot pull new work; the case remains effectively "stuck," preventing efficient release of capacity upstream and causing intermittent idleness at the initial stage despite a growing queue. Second, the baseline capacities of all three ResourcePools are set to 1, meaning only one staff member per stage. Under a workload exceeding 20,000 arrivals, a single saturated expert reviewer becomes the dominant constraint, throttling the entire workflow and propagating congestion upstream (Figures 2 and 9).

To assess improvements, a sensitivity analysis was conducted by changing staffing levels (ResourcePool capacities) using the model's slider controls, and five scenarios were evaluated relative to the baseline. In Scenario 1, the capacities of InitialStaff and ExpertStaff were increased from 1 to 5, and TestStaff was increased from 1 to 3. The resulting statistics and dashboard outputs are presented in Figures 10–13. This scenario reduces mean cycle time from 28,270 to 23,937 minutes (Figure 21), corresponding to an improvement of about 15% in overall throughput speed. The initial review queue length decreases from approximately 18,700 to 15,000 (Figures 11 and 13), indicating higher responsiveness to incoming requests although the remaining backlog suggests that the system is not fully stable yet. Test-stage utilization increases from 21% to 37% (Figure 13), reflecting that more work reaches the testing stage once upstream capacities are expanded.

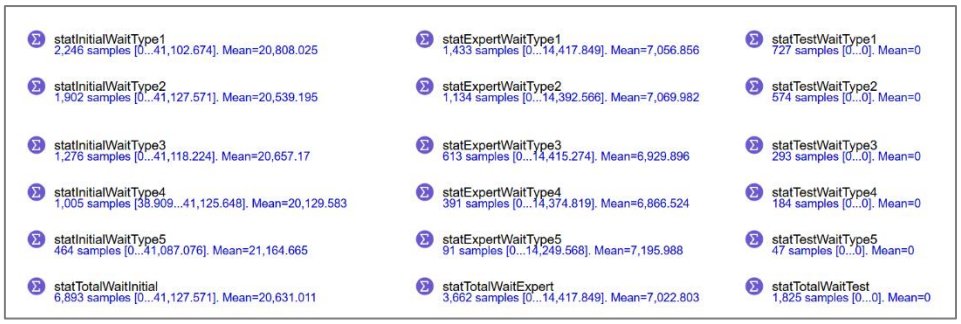

Figure 10
Simulation Results For Scenario 1 – Statistics Elements Output

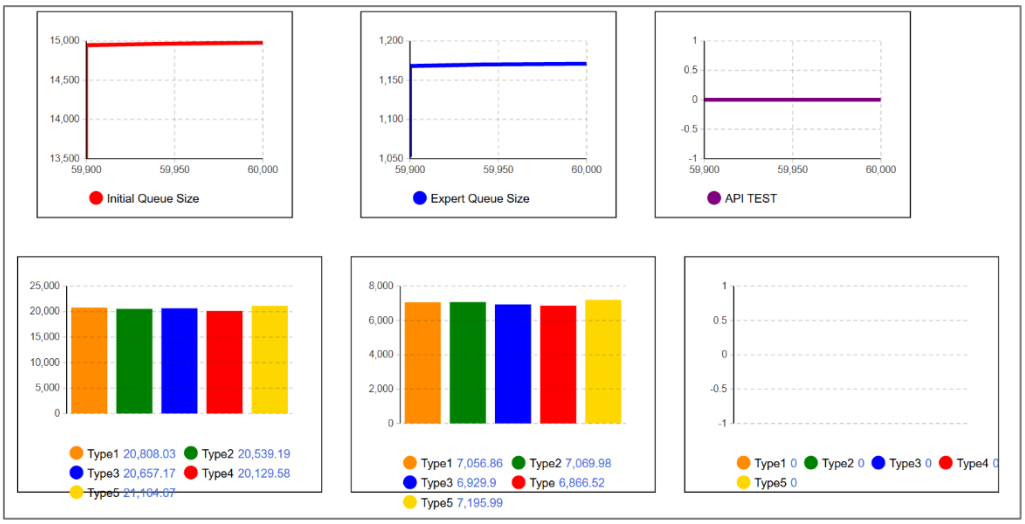

Figure 11
Scenario 1 – Queue Performance Plots (Queue Sizes and Waiting Indicators)

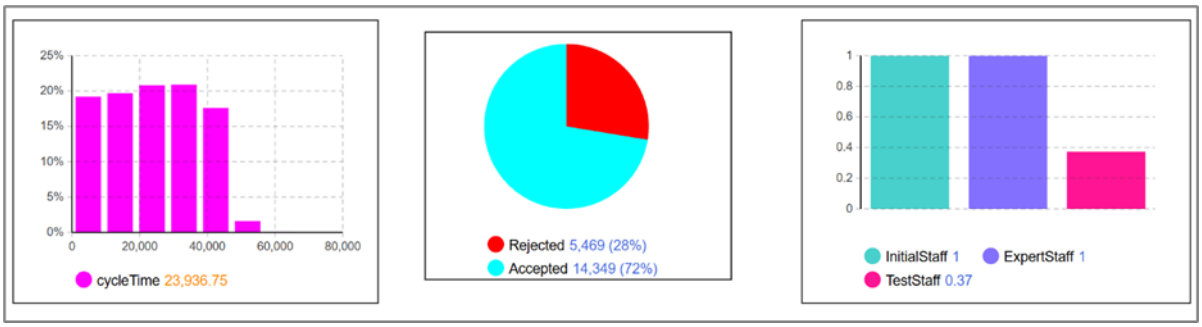

Figure 12
Scenario 1 – Analytical Charts and KPI Visualizations

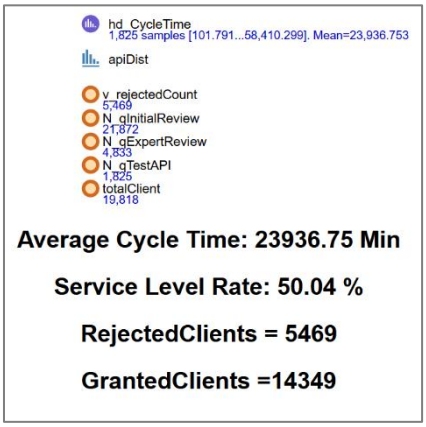

Figure 13
Scenario 1 – Summary of Simulation Results (Key KPIs)

In Scenario 2, staffing in the two sensitive stages was increased further (InitialStaff and ExpertStaff from 5 to 7) while TestStaff was reduced (3 to 2) to reallocate capacity from a less constrained stage to higher-pressure stages. Results are reported in Figures 14–17. The initial review queue length decreases further from about 15,000 to 13,000 (Figures 15 and 17). However, the expert review queue length increases from about 1,170 to 1,570 (Figure 25), indicating that the bottleneck shifts toward the expert review stage, as the first stage, pushes more cases downstream. Despite reducing TestStaff, test utilization remains about 37% (Figure 17), suggesting that part of the testing capacity in Scenario 1 was redundant.

Figure 14
Simulation Results For Scenario 2 – Statistics Elements Output

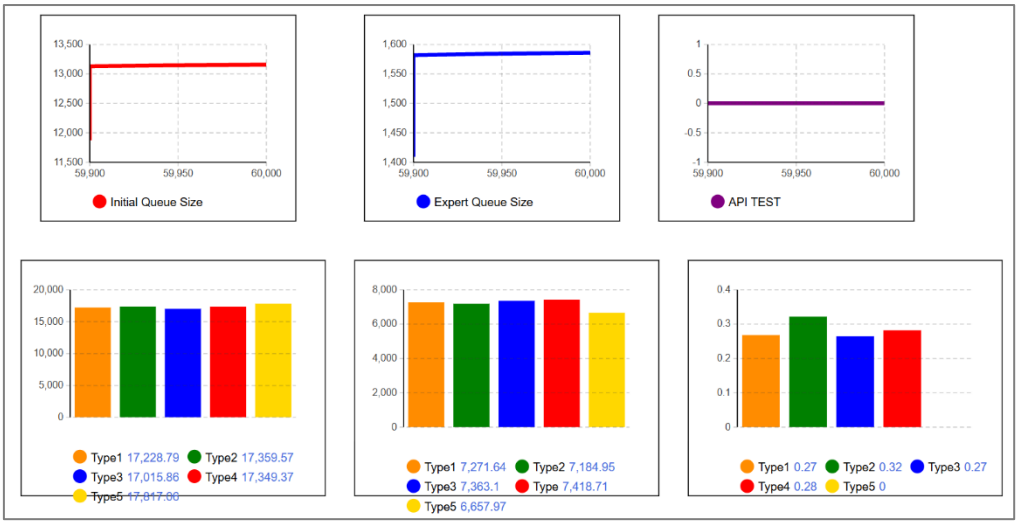

Figure 15
Scenario 2 – Queue Performance Plots (Queue Sizes and Waiting Indicators)

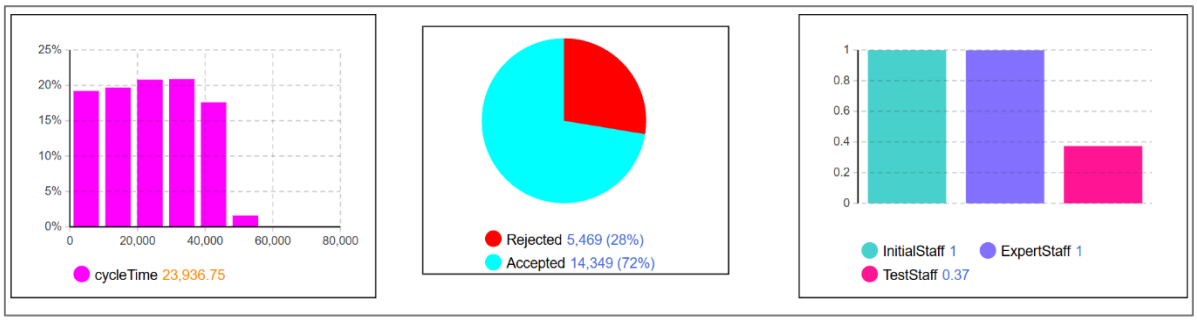

Figure 16
Scenario 2 – Analytical Charts and KPI Visualizations

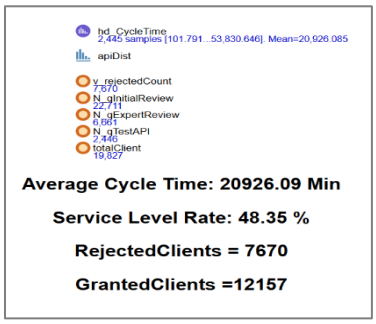

Figure 17
Scenario 2 – Summary of Simulation Results (Key KPIs)

In Scenario 3, both InitialStaff and ExpertStaff were increased to 10 (the defined maximum), while TestStaff remained at 2. The results (Figures 18–21) show a substantial improvement: the mean cycle time decreases to 19,310 minutes (Figure 21), a 32% improvement relative to baseline. The initial review queue length drops from about 18,700 to 10,500 (Figures 19 and 21), indicating that the system begins to overcome the arrival rate and actively deplete backlog. Meanwhile, the expert review queue length increases (from 1,570 in

Scenario 2 to 2,200 in Scenario 3; Figure 21), consistent with higher upstream throughput transferring pressure to the second stage.

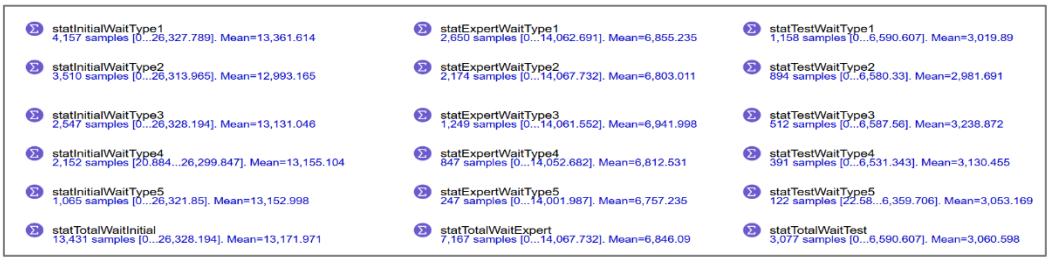

Figure 18
Simulation Results for Scenario 3 – Statistics Elements Output

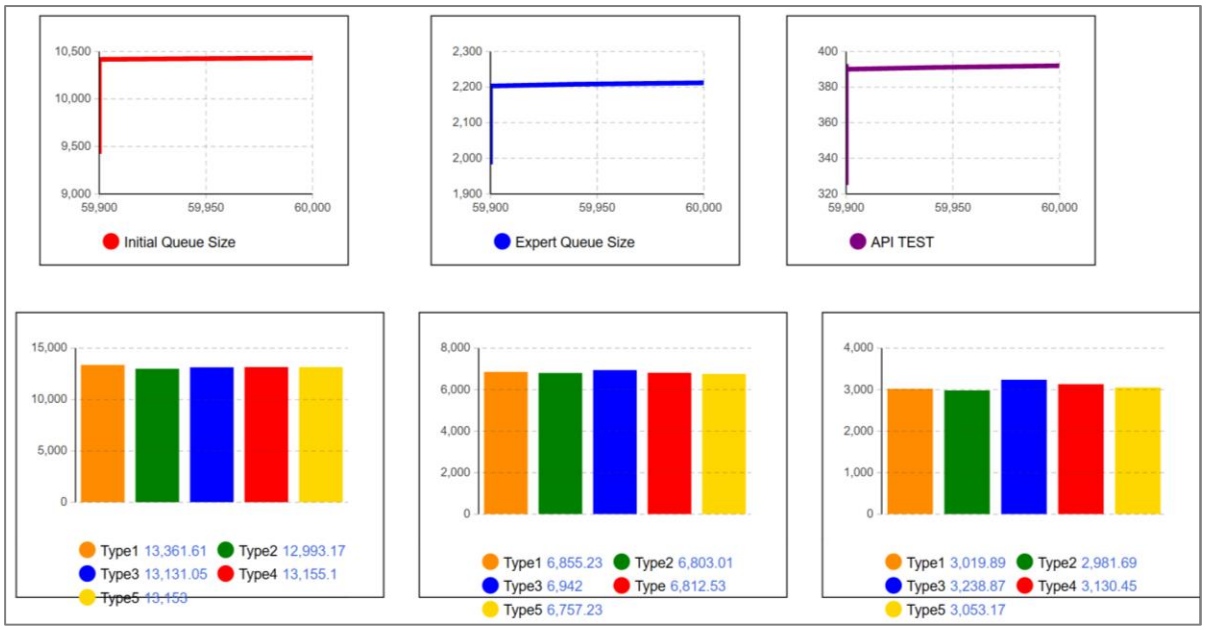

Figure 19
Scenario 3 – Queue Performance Plots (Queue Sizes and Waiting Indicators)

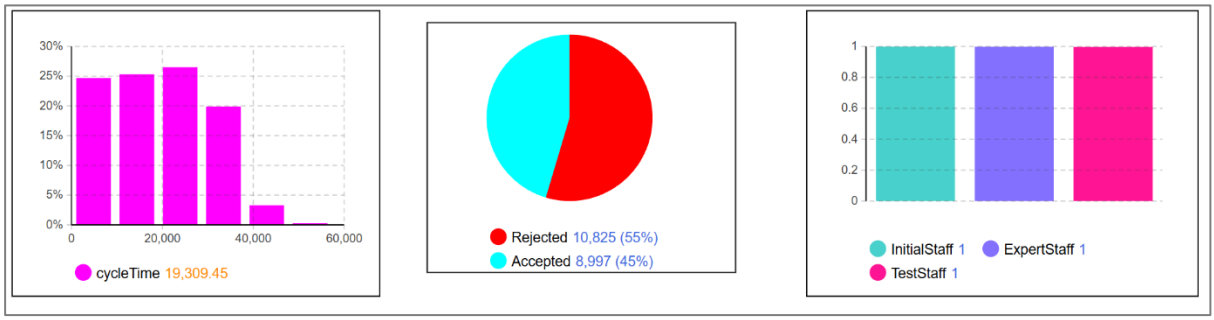

Figure 20
Scenario 3 – Analytical Charts and KPI Visualizations

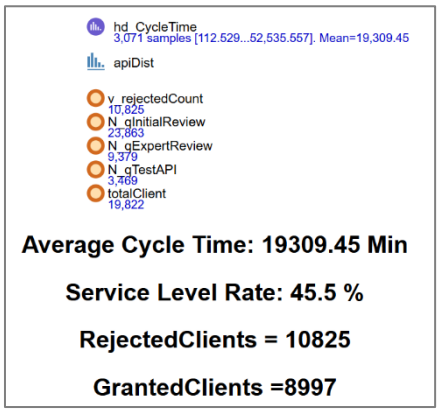

Figure 21
Scenario 3 – Summary of Simulation Results (Key KPIs)

In Scenario 4, InitialStaff and ExpertStaff were kept at 10, and TestStaff was increased from 2 to 5 to examine the effect of adding capacity to a non-critical stage. Outputs are presented in Figures 22–25. This configuration achieves the lowest mean cycle time among the tested scenarios (17,996 minutes; Figure 25), corresponding to about a 36% reduction relative to baseline. However, the initial review queue length remains approximately 10,400 (Figure 25), indicating that once the upstream stages reach saturation, adding more staff in downstream testing does not materially reduce the entry backlog. Test utilization decreases to 45% (Figure 25), suggesting that the additional testing capacity becomes partially idle and does not translate proportionally into system-wide gains.

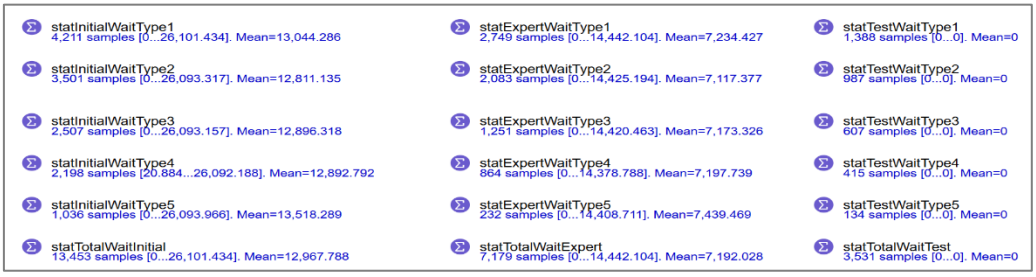

Figure 22
Simulation Results for Scenario 4 – Statistics Elements Output

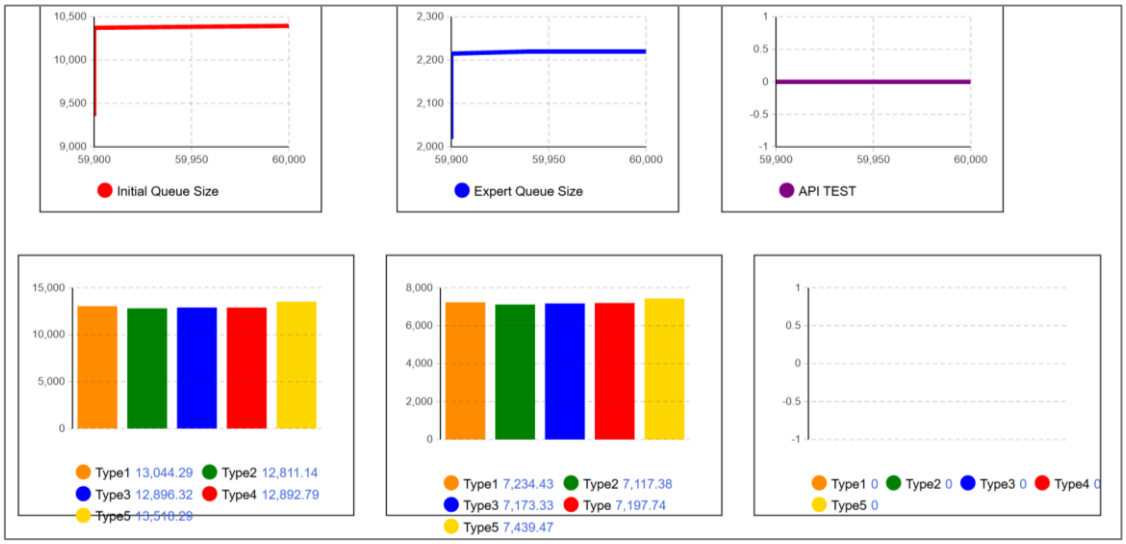

Figure 23
Scenario 4 – Queue Performance Plots (Queue Sizes and Waiting Indicators)

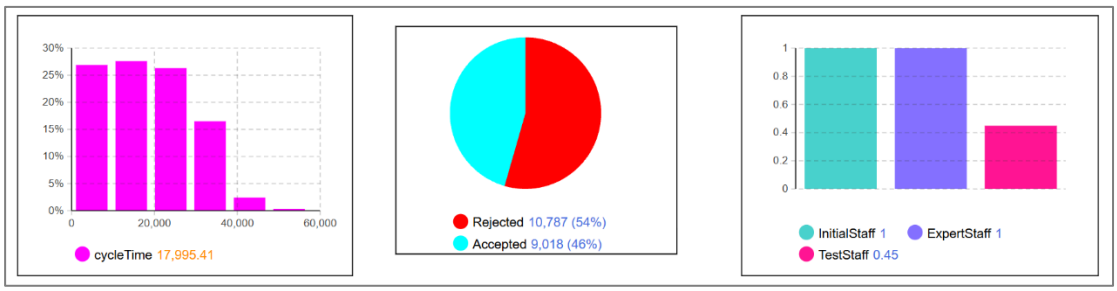

Figure 24
Scenario 4 – Analytical Charts and KPI Visualizations

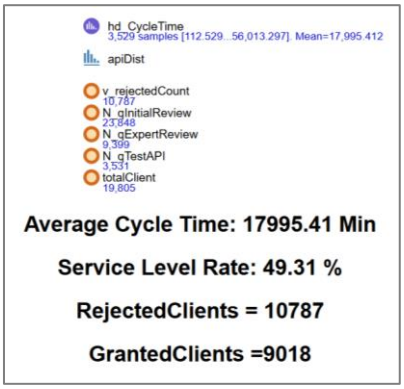

Figure 25
Scenario 4 – Summary of Simulation Results (Key KPIs)

In Scenario 5, InitialStaff and ExpertStaff remained at 10, while TestStaff was reduced to 3 to seek a more balanced operating point. Results are presented in Figures 26–29. The mean cycle time becomes 18,821 minutes (Figure 29), only about 4% higher than Scenario 4, while still maintaining strong improvement versus baseline. Importantly, test utilization increases to 74% (Figure 29), indicating a more efficient allocation of testing resources without substantial

loss in end-to-end performance. The initial review queue length remains stable at approximately 10,400 (Figure 29), reinforcing that further reduction of the entry backlog would require changes at the initial stage itself (e.g., process redesign or increased effective service capacity), rather than additional downstream staffing.

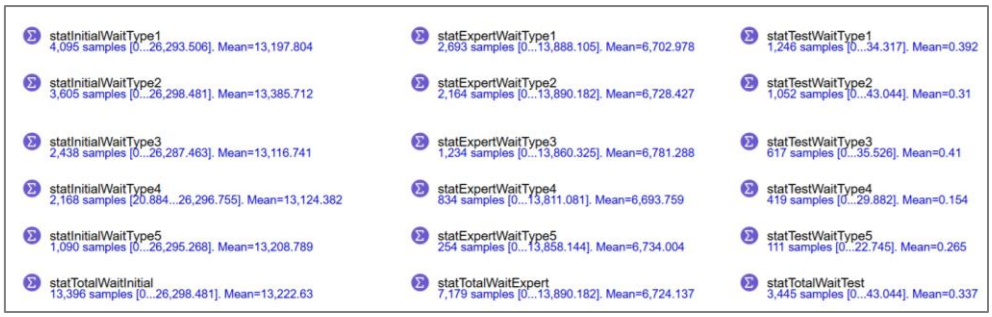

Figure 26
Simulation Results for Scenario 5 – Statistics Elements Output

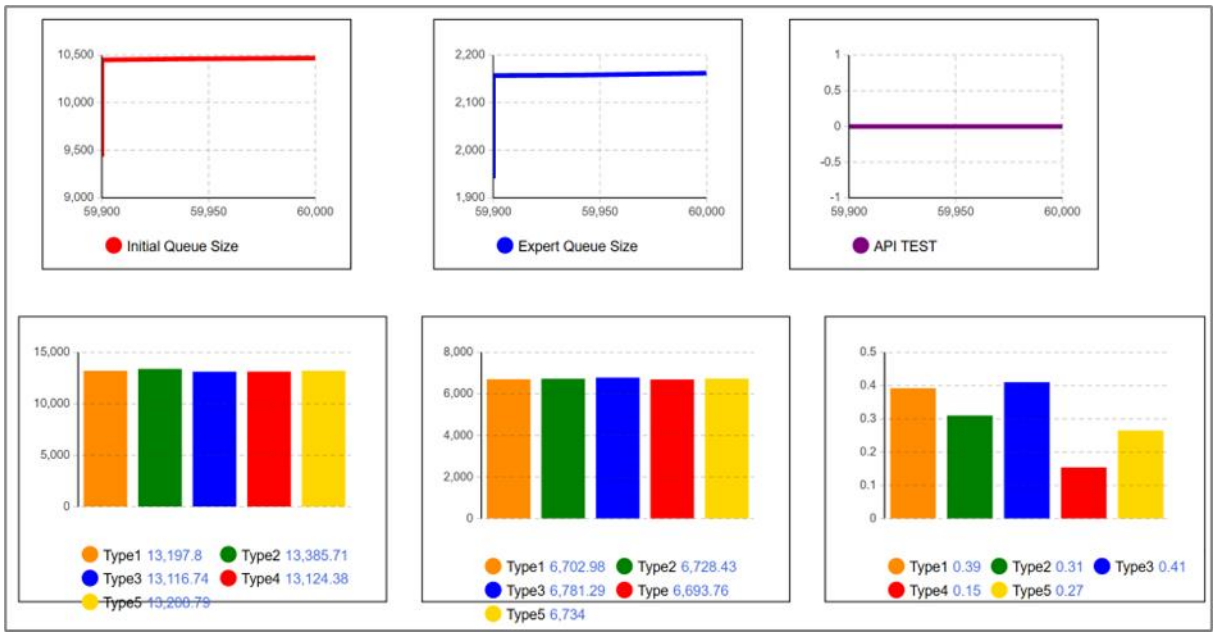

Figure 27
Scenario 5 – Queue Performance Plots (Queue Sizes and Waiting Indicators)

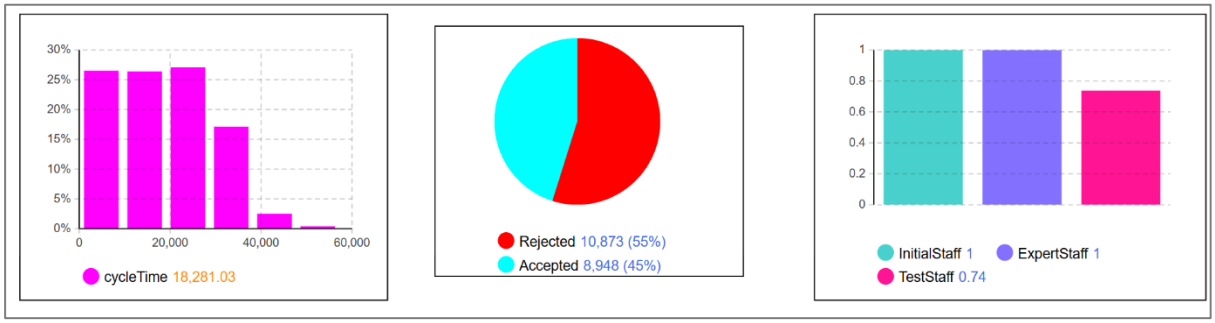

Figure 28
Scenario 5 – Analytical Charts and KPI Visualizations

Figure 29
Scenario 5 – Summary of Simulation Results (Key KPIs*)*

Discussion and Conclusion

This study used discrete-event simulation as a diagnostic and decision-support approach to evaluate a multi-stage API access granting process and to identify a more effective workforce configuration. The motivation was to analyze the system's dynamic response to fluctuating demand and stochastic service behavior, and to reduce reliance on intuition-based operational decisions that can lead to inefficient use of human resources. The developed model is designed to address practical managerial questions such as: how long clients wait in the initial review, expert review, and testing queues; whether any stage becomes a bottleneck due to limited capacity; and what portion of clients experience excessive delays, as reflected in the cycle-time distribution.

The baseline run indicates a severely congested system. The dashboard reports an Average Cycle Time of about 28,269 minutes (approximately 19 days), which is consistent with an unstable regime in which requests accumulate and remain in the system for extended periods. In the same baseline configuration, the Service Level Rate is 48.79%, meaning that fewer than half of arrivals meet the model's service-level criterion. At the same time, 94% of requests are ultimately accepted, indicating that the dominant problem is not rejection or failure to complete processing, but rather excessive waiting and long end-to-end delays. Operationally, this implies that high acceptance does not translate into satisfactory client experience when the time-to-grant is highly large.

A key diagnostic insight is the identification of a bottleneck mechanism that explains the apparently contradictory baseline behavior: the initial review queue can be long while initial-stage utilization remains low. In the baseline results, Initial Staff utilization is 16%, while Expert Staff utilization is about 99%. This pattern is consistent with downstream blocking: when expert review is saturated, work items cannot move forward, which prevents upstream resources from continuously starting new items even if many requests are waiting. In addition, the baseline staffing assumption (capacity set to 1 for each of the three Resource Pools) makes the system highly sensitive to demand; under a workload exceeding 20,000 incoming requests, a single near-saturated expert reviewer becomes a dominant constraint that propagates congestion upstream. The scenario experiments confirm that bottlenecks are not fixed; instead, they shift across stages as capacity is adjusted. When the two early stages were strengthened (e.g., increasing staffing in the initial and expert layers), the system began to reduce overall delay, but the pressure also moved downstream and manifested as increased accumulation in later queues. This behavior highlights an important systems-level implication: improving one station in a multi-stage service process can transfer congestion to the next station rather than

eliminating it; therefore, improvements must be evaluated using a combined view of waiting time, queue length, and resource utilization, not by focusing on a single queue in isolation.

Across the tested configurations, the results indicate that expanding capacity in critical stages yields measurable reductions in mean cycle time, while over-expanding a non-critical stage can create avoidable idle capacity. In particular, Scenario 4 demonstrates that increasing testing capacity while keeping the upstream stages fixed at their maximum does not materially reduce the initial queue (which remains near the same level), while it reduces testing utilization. This indicates that once upstream capacity becomes the binding constraint, further downstream staffing provides diminishing returns and may not be cost-effective.

Considering the combined evidence from cycle time, queue behavior, and utilization, Scenario 5 emerges as the most balanced configuration among the evaluated scenarios. In this scenario, staffing in the initial and expert review stages is kept at the maximum tested level (10 each), while testing staff is set to 3. The mean time in system under Scenario 5 is 18,821 minutes, which reflects a substantial improvement relative to the baseline mean of 28,270 minutes, reported in the scenario comparison table. Importantly, Scenario 5 achieves a higher utilization of the testing workforce (74%) than Scenario 4 (45%), with only a modest increase in mean cycle time compared with Scenario 4 (18,821 minutes vs. 17,996 minutes). From an operational planning perspective, this suggests that Scenario 5 provides a more efficient allocation of downstream capacity while preserving most of the end-to-end time benefit gained in Scenario 4.

Based on these findings, the study supports several operational recommendations. First, when persistent backlog and long cycle times are observed, increasing staffing in the two early layers (initial review and expert review) is generally more impactful than adding capacity primarily to testing. Second, a practical extension is to implement dynamic cross-allocation, where a portion of testing staff temporarily supports initial review when the entry queue exceeds a management-defined threshold, improving resilience during demand spikes without permanently increasing headcount. Third, the initial review stage can be strengthened through intelligent pre-screening, where rule-based or AI-assisted checks filter clearly non-compliant submissions early, reducing unnecessary downstream load.

This study has several limitations. First, while the arrival process is data-driven, the empirical event log used is not an Open Banking operational log; therefore, the mapped arrival dynamics may not fully reflect bank-specific seasonality, policy-driven surges, or client behavior in real Open Banking onboarding environments. Second, key service-time assumptions were represented using simplified parametric distributions and typedependent ranges; richer distribution fitting by activity, time-of-day, and reviewer role (or direct calibration from real service logs) could improve realism. Third, staff are modeled as fixed-capacity resources without explicitly capturing heterogeneity among reviewers (skill differences), learning effects, shift calendars, multitasking, interruptions, or cross-training—factors that can materially affect waiting times and bottlenecks.

Fourth, exogenous drivers (e.g., regulatory change, outages, or upstream documentation quality shocks) are outside the model boundary, which may understate variability under stress conditions. Finally, the scenario analysis evaluates operational KPIs but does not include an explicit cost function or formal optimization; future work can integrate cost–service trade-offs and use optimization experiments to search staffing policies under service-level constraints.

In conclusion, the simulation acts as a virtual laboratory for diagnosing bottlenecks and evaluating workforce decisions in heavy queueing environments. The baseline results demonstrate that high acceptance alone does not ensure effective service when cycle times are excessive, and the scenario results highlight that bottlenecks can migrate across stages as capacity changes. Among the tested alternatives, Scenario 5 provides the most robust balance

between improved end-to-end performance and the efficient utilization of the testing workforce, making it the recommended operational configuration within the scope of this study.

References

Adeleke, A. G., Sanyaolu, T. O., Efunniyi, C. P., Akwawa, L. A., & Azubuko, C. F. (2024). API integration in FinTech: Challenges and best practices. *International Journal of Financial Technology*.

Babina, T., Bahaj, S., Buchak, G., De Marco, F., Foulis, A., Gornall, W., Mazzola, F., & Yu, T. (2025). Customer data access and fintech entry: Early evidence from open banking. *Journal of Financial Economics*, *169*, 103950.

Banks, J., Nicol, D., Carson, J., & Nelson, B. (2013). *Discrete-event system simulation*. Pearson Deutschland.

Barbosa Casolaro, A. M., Rauber, G. N., & de Lima, U. S. M. (2025). Open banking: A systematic literature review. *Journal of Banking Regulation*, *26*, 340-355. https://doi.org/10.1057/s41261-024-00262-x.

Bocciarelli, P., & D'Ambrogio, A. (2024). Simulation-based predictive process mining with ebpmn: Methods, challenges and opportunities. In *2024 Annual Modeling and Simulation Conference (ANNSIM)*.

Briones de Araluze, G., & Cassinello Plaza, E. (2022). Open banking: A bibliometric analysis-driven definition. *PLOS ONE*, *17*(10), e0275496. https://doi.org/10.1371/journal.pone.0275496.

Camargo, M., Dumas, M., & González-Rojas, O. (2020). Automated discovery of business process simulation models from event logs. *Decision Support Systems*, *134*, 113284.

Camargo, M., Dumas, M., & González-Rojas, O. (2022). Learning accurate business process simulation models from event logs via automated process discovery and deep learning. In *International Conference on Advanced Information Systems Engineering.*

Casolaro, A. M. B., Rauber, G. N., & de Lima, U. S. M. (2025). Open banking: A systematic literature review. *Journal of Banking Regulation*, *26*(3), 340-355.

Che, E., Dong, J., & Namkoong, H. (2024). Differentiable discrete event simulation for queuing network control. *arXiv preprint arXiv:2409.03740*.

Gomber, P., Kauffman, R. J., Parker, C., & Weber, B. W. (2018). On the FinTech revolution: Interpreting the forces of innovation, disruption, and transformation in financial services. *Journal of Management Information Systems*, *35*(1), 220-265.

Gounari, M., Stergiopoulos, G., Pipyros, K., & Gritzalis, D. (2024). Harmonizing open banking in the European Union: An analysis of PSD2 compliance and interrelation with cybersecurity frameworks and standards. *International Cybersecurity Law Review*, *5*, 79-120. https://doi.org/10.1365/s43439-023-00108-8.

Hossain, M. A. (2025). Investigating the cybersecurity implications of open banking and application programming interfaces (APIs) in the financial sector. *SSRN 5207072*.

Hota, A. (2022). Securing API ecosystems in digital banking transformation.

Law, A. M. (2014). *Simulation modeling and analysis (Mcgraw-hill series in industrial engineering and management)* (5th ed.). McGraw Hill.

Liao, C.-H., Guan, X.-Q., Cheng, J.-H., & Yuan, S.-M. (2022). Blockchain-based identity management and access control framework for open banking ecosystem. *Future Generation Computer Systems*, *135*, 450-466.

Lin, X., Zhang, S. S., & Zachariadis, M. (2025). Open data and API adoption of US banks. *Journal of Financial Intermediation*, *63*, 101162.

López-Pintado, O., & Dumas, M. (2023). Discovery and simulation of business processes with probabilistic resource availability calendars. In *2023 5th International Conference on Process Mining (ICPM)*.

Modesti, P., Freitas, L., Shotomiwa, Q., & Almehrej, A. (2025). Security analysis of the open banking account and transaction API protocol. *Cyber Security and Applications*, *3*, 100097.

Mohammed, A. (2024). Open banking and APIs: Research on how open banking frameworks and APIs are reshaping the financial ecosystem. *International Journal of Advances in Engineering and Management*, *7*, 770-784.

Muqorobin, M. M., Anggraini, A., Rahmawati, A. D., Yohanes, D., & Ifkarina, F. D. (2021). Pengaruh open banking berbasis open API terhadap eksistensi perbankan. *MAKSIMUM: Media Akuntansi Universitas Muhammadiyah Semarang*, *11*(2), 75-84.

Niankara, I., Hassan, H. I., Traoret, R. I., & Islam, A. R. M. (2025). Consumer savings and digital remittance in open banking: Insights from bibliometric and geospatial econometric analysis. *Human Behavior and Emerging Technologies*, *2025*(1), 9352257.

Ojehomon, O. G., Cichorska, J., & Michnik, J. (2026). Cyber risk management of API-enabled financial crime in open banking services. *Entropy*, *28*(2), 163.

Pandey, D., bajpai, P., & Kumar, D. (2025). Cybersecurity in Open Banking: Securing APIs and protecting customer data in a connected world. *Economic Sciences*, *21*, 296-301. https://doi.org/10.69889/pz6qhp04.

Riad, K., & Elhoseny, M. (2022). A blockchain-based key-revocation access control for open banking. *Wireless Communications and Mobile Computing*, *2022*(1), 3200891.

Robinson, S. (2005). Discrete-event simulation: From the pioneers to the present, what next? *Journal of the Operational Research Society*, *56*(6), 619-629. https://doi.org/10.1057/palgrave.jors.2601864.

Rozinat, A., Mans, R. S., Song, M., & van der Aalst, W. M. (2009). Discovering simulation models. *Information Systems*, *34*(3), pp.305-327.

Septiyanto, A. F., Sarno, R., & Sungkono, K. R. (2022). Identifying bottlenecks of hierarchical process model by using discrete event simulation and process mining. In *2022 IEEE 8th International Conference on Computing, Engineering and Design (ICCED).*

Shacheendran, V., Lukose, A., Josiah, J., Joseph, D., & Jeena, J. (2025). The rise of open banking: A comprehensive analysis of research trends and collaborative networks. *International Journal of Economics and Financial Issues*, *15*(1), 295.

Sharma, V. (2025). Open Banking with AI. *International Journal of Artificial Intelligence*, *5*(09), 721-728. https://www.academicpublishers.org/journals/index.php/ijai/article/view/6216.

Van Der Aalst, W. (2012). Process mining: Overview and opportunities. *ACM Transactions on Management Information Systems (TMIS)*, *3*(2), 1-17.

Veldurthi, A. K. (2025). The API economy: Driving fintech innovation through open banking and embedded finance. *World Journal of Advanced Research and Reviews*, *26*(2).

Wang, T., Song, Y., & Hong, J. (2023). Fast approximation to discrete-event simulation of markovian queueing networks. In *2023 Winter Simulation Conference (WSC)*.