International Journal Of Health, Engineering And Technology (IJHET) *E-ISSN 2829 - 8683*
Volume 4, Number 3, September 2025, *Page. 319 - 329*
Email : editorijhess@gmail.com

Optimization of Goods Tracking and Inventory Systems at PT Integrasi Data Nusantara Using Genetic Algorithms and Functional TestingTHE EFFECT OF AREN

Tubagus Adhitya Permana[1], Fadillah Said[2], Sri Lestari[3], Kiki Setiawan[4]

[1,2,3,4)] Cipta Karya Informatika College of Computer Science
Email: [1]aditt7116@gmail.com, [2]said.fadillah@stikomcki.ac.id, [3]sri.lestari1203@gmail.com, [4]ki2djoaz@gmail.com

Abstract

Inventory management is a critical aspect of business operations, but at PT Integrasi Data Nusantara, the process of determining optimal stock levels and reorder points is still done manually, leading to risks of overstocking or stockouts. This results in high operational costs and potential dissatisfaction among training participants due to a mismatch between supply and dynamic demand. This study aims to optimize the goods tracking and inventory system by using a Genetic Algorithm to address these inefficiencies. The research was conducted at PT Integrasi Data Nusantara using historical training demand data and associated inventory cost data. The methodology includes designing an optimization system, implementing a genetic algorithm with a chromosome structure (ROP, ROQ), a fitness function that minimizes total costs, and evolutionary operators (selection, crossover, mutation). The research results show that the developed system is effective in recommending optimal inventory policies, successfully eliminating all stockout incidents, and significantly improving cost efficiency compared to manual methods. The conclusion of this study is that the implementation of a Genetic Algorithm can be an adaptive and functional solution to support more accurate, efficient, and automated inventory decision-making.

***Keywords**: Genetic Algorithm, Optimization, Inventory Management, Reorder Point, Reorder Quantity*

INTRODUCTION

Effective inventory management is a crucial aspect of business operations, particularly for companies that handle a diverse range of goods and require high efficiency to avoid excessive costs or stock shortages. At PT Integrasi Data Nusantara (PT IDN), an information technology company specializing in training and consulting, the challenge of managing inventory for training materials, such as modules, apparel, pens, and notebooks, becomes increasingly complex. The need for accurate, real-time stock information and efficiency in determining optimal stock levels and reorder points is pressing. An effective and efficient inventory system not only ensures smooth training sessions and operational continuity but also contributes to optimizing operational costs and enhancing customer satisfaction (Rupilele & Lahallo, 2024). PT IDN faces challenges in determining the optimal quantity of each inventory item to hold, as well as the most efficient reorder points. This is critical to prevent overstocking, which leads to high storage costs and the risk of obsolete items, and stockouts, which can cause delays or cancellations of training programs and result in customer dissatisfaction (Witari et al., 2021).

The conventional method of manual estimation, based on historical data and experience, is no longer sufficient to cope with the dynamic fluctuations in demand and market conditions (Sari et al., 2020). These manual processes are prone to human error, time-consuming, and fail to adapt quickly to sudden changes. This often leads to sub-optimal inventory decisions that directly impact operational cost efficiency and the smooth execution of training activities. Consequently, there is a clear need for a more advanced and automated approach to solve this complex inventory optimization problem. Genetic Algorithms (GAs) provide a promising solution. As a metaheuristic optimization method that mimics the process of natural selection, GAs are highly effective in finding optimal solutions for complex problems with large search spaces and multiple constraints, such as the simultaneous determination of Reorder Point (ROP) and Reorder Quantity (ROQ) (Syam et al., 2020; Utama et al., 2023).

Despite the use of basic systems for recording inventory, PT IDN's process for determining stock levels remains manual. This presents three primary issues. First, the determination of ROP and ROQ is sub-optimal because it fails to consider all relevant factors, including holding, ordering, and stockout costs (Laoli et al., 2022; Utami et al., 2024). Second, the company is at a high risk of both overstocking and stockouts. Overstocking results in high storage costs and potential obsolescence, while stockouts can disrupt training schedules and lead to customer dissatisfaction (Sari et al., 2020). Third, the current inventory management method struggles to adapt to dynamic fluctuations in training demand, leading to slow and inaccurate responses to real needs and a potential loss of business opportunities (Syam et al., 2020).

These challenges collectively lead to operational inefficiency and threaten the seamless execution of training programs at PT IDN. Therefore, a comprehensive system is needed that not only records data but also possesses the analytical capability to automatically optimize inventory levels. The implementation of a GA-based system is expected to address these inefficiencies by providing accurate and efficient recommendations, empowering the company to make faster and more effective decisions. This research aims to bridge the gap between traditional inventory recording systems and the need for an intelligent optimization system.

Based on the problems outlined above, this study aims to implement a Genetic Algorithm to optimize the simultaneous determination of inventory levels and reorder points at PT Integrasi Data Nusantara and validate the system's effectiveness through functional testing. The urgency of this research lies in PT IDN's critical need to transition from error-prone manual methods to an automated system. This transition will significantly reduce operational costs, mitigate the risks of overstocking and stockouts, and enhance participant satisfaction. The novelty of this research lies in its specific focus on optimizing inventory for training materials, where a GA is used to simultaneously optimize ROP and ROQ using a fitness function that minimizes total inventory costs. This comprehensive approach is not widely explored in this specific context. Furthermore, this study explicitly emphasizes end-to-end functional testing to ensure the proposed solution is not just theoretically sound but also practically implementable and effective in a real-world operational environment. The findings are expected to serve as a valuable reference for other companies and future research on applying optimization techniques to solve complex inventory problems.

RESEARCH METHODS

Research Type and Method

This research employs a quantitative approach with a case study design, focusing on the development and optimization of an information system. The quantitative method is used to analyze numerical data, such as historical demand and inventory costs, to produce objective and measurable results (Waruwu et al., 2025). The case study approach is suitable for in-depth analysis of a specific phenomenon within a real-world context, which in this case is the inventory management process at PT Integrasi Data Nusantara (Sugiyono, 2022). This method allows for a comprehensive understanding of the operational challenges and the effectiveness of the proposed GA-based solution (Emzir, 2021). The research also falls under applied research, as it seeks to solve a specific practical problem faced by the company, thereby providing tangible and direct contributions (Sari et al., 2020).

Data and Data Analysis

The research utilizes several types of data relevant to the inventory management problem at PT Integrasi Data Nusantara. The data serves as the foundation for designing and implementing the Genetic Algorithm. Key data points include:

A. Item Data: Details of each training-related item managed by the company, such as item name (e.g., training modules, T-Shirtts, pens, notebooks, and tote bags) and unique SKU codes.
B. Historical Demand Data: Historical records of demand or sales for each item, which are essential for forecasting future needs.
C. Cost Data: Inventory-related costs, including unit holding costs, ordering costs, and estimated stockout costs.

Data analysis in this study is performed through the core mechanism of the Genetic Algorithm. The GA processes the collected data to find the optimal solution by iteratively evolving a population of potential solutions. The primary analytical instrument is the GA itself, which is configured with specific

parameters to minimize total inventory costs (Utama et al., 2023). This approach is more sophisticated than traditional methods and is well-suited for handling the complex interdependencies of inventory variables (Laoli et al., 2022).

Population and Sample

In this case study, the population consists of the entire set of training-related items managed by PT Integrasi Data Nusantara. The sample for this study is the comprehensive dataset collected from the company, comprising historical demand and cost data for a specific period. Since this is an in-depth case study on a single entity, the entire relevant data set of the company is used, as opposed to sampling from a larger, more generalized population. This approach, as noted by research methodologists, is valid for case studies where the focus is on a single, well-defined entity rather than broad generalizations (Sugiyono, 2022). The goal is not statistical generalization but a deep, contextual understanding of the problem and the validation of the solution within this specific operational environment.

Research Procedures

The research procedure is structured into a logical, systematic flow to achieve the stated objectives. The following stages will be carried out sequentially:

1. Problem Identification and Literature Review: The research begins by a deep dive into the research problem, which is the sub-optimal manual inventory management at PT IDN. A comprehensive literature review is then conducted to understand existing theories, methodologies, and the research gap, particularly concerning the application of Genetic Algorithms for inventory optimization (Sari et al., 2020; Syam et al., 2020).
2. Needs Analysis: This stage involves collecting detailed data on items, historical demand, and operational constraints at PT IDN. The aim is to define the functional and non-functional requirements of the proposed system and identify the specific constraints that the GA model must consider.
3. System Design: This phase focuses on designing the GA-based optimization system. The design includes:
    A. Chromosome Representation: A chromosome, representing a potential solution, is defined as a collection of genes. Each gene contains information on the optimal quantity (Q) and reorder point (R) for a specific item. The length of the chromosome depends on the number of items to be optimized. This process of encoding solutions into chromosomes forms the initial population for the GA.
    B. Fitness Function Evaluation: A fitness function is designed to measure the performance of each chromosome. Its primary objective is to minimize the total inventory cost, which includes holding, ordering, and stockout costs. The function also incorporates a penalty for any violations of constraints, such as exceeding warehouse capacity or failing to meet a desired service level (Utami et al., 2024).
    C. Selection: This process involves selecting the fittest individuals (chromosomes) from the current population to be carried forward to the next generation. Methods like tournament selection or roulette wheel selection can be used to ensure that chromosomes with better fitness values have a higher chance of being selected.
    D. Crossover: As a key genetic operator, crossover combines genetic material from two parent chromosomes to produce new offspring. One-point crossover will be implemented, where two parents are split at a single point and the segments are swapped to create new children with a combination of parental traits. This operator drives the algorithm's exploration of the solution space.
    E. Mutation: Mutation introduces random changes to the chromosomes to maintain genetic diversity and prevent the algorithm from getting stuck in local optima. A low mutation rate is applied to randomly alter genes within a chromosome, potentially leading to a new, better solution in a later iteration (Salman et al., 2025).
4. System Implementation: The system will be built based on the design specifications. The implementation will focus on three key functionalities: (1) calculating and recommending optimal inventory quantities, (2) determining efficient reorder points, and (3) enabling an adaptive inventory strategy that can respond to changes in demand.
5. Testing and Analysis: Functional testing will be the primary method for validating the system's performance. The goal is to verify that all system features, including data input, optimization

calculations, and recommendation display, operate as expected. The analysis will compare the system's output (optimal costs, ROP, ROQ) with the results from the manual method to demonstrate its effectiveness in reducing costs and operational risks (Nugroho & Hutahaean, 2025).

6. Results and Conclusion: The final stage involves presenting, interpreting, and summarizing the results of the optimization and system validation. Conclusions will be drawn regarding the effectiveness of the GA-based system, and recommendations for future research will be provided based on the study's findings and limitations.

RESULTS AND DISCUSSION

Historical Data Processing Training

This stage outlines the transformation of raw data, derived from both historical records and projected future demand, into a structured format suitable for use in the genetic algorithm model. Historical training registration data from January to June 2025 (history-training-jan-juni2025.csv) was read and processed to provide an overview of past demand. The process included data cleaning, such as removing irrelevant characters and standardizing the format of training names, participant names, and T-Shirtt sizes, as well as extracting key information by generating unique lists of training sessions and T-Shirtt sizes previously ordered. The processed data revealed a total of 1,892 participants, corresponding to 1,892 T-Shirtts requested.

Table 1. T-Shirtt Size Distribution from Historical Data

Clothe Size	Number of Participants	Percentage
L	819	43.30%
XL	609	32.20%
XXL	311	16.50%
XXXL	153	8.10%

Based on historical data and the inclusion of common items such as pens, notebooks, and tote bags, the total number of unique items that need to be managed by the system amounts to 573. This consists of 456 types of training T-Shirtts (covering various training programs and sizes), 114 types of training modules, and 3 common items, namely pens, notebooks, and tote bags.

Future Demand Data Processing

Future demand data for the August and September 2025 period (data_2_bulan_kedepan.csv) is also processed to serve as the main input for the optimization model. This process involves data cleaning and consolidation, where the CSV file is read and columns such as "training" and "location" are standardized. The "Jadwal" (Schedule) column is processed to obtain specific training dates. The estimated number of participants is extracted from the "peserta" (participants) column, which is in an X/Y format, with the Y value (maximum class capacity) being used as the demand forecast for each training session. This is done because the Y value represents the ideal target or maximum number of participants that must be prepared for, making it a relevant input for inventory planning. A summary of the estimated monthly demand is obtained from this data analysis.

Table 2. Estimated Demand for Training Participants (August-September 2025)

Month	Total Estimated Participants
Agustus 2025	734
September 2025	759

Identification of Inventory Items, Cost of Goods Sold (COGS), and Lead Time

Based on the processed data, the system must manage a total of 573 unique inventory items. This figure results from the combination of each unique T-Shirtt by training type and size, training modules by program, and general items. The unique inventory requirements consist of 456 types of training T-Shirtts (across various programs and sizes), 114 types of training modules, and 3 common items (pen, notebook, and tote bag). An illustrative table is provided to summarize a subset of these inventory items along with their assumed COGS and lead times. The complete list is incorporated into the algorithm to represent each gene within the chromosome.

Table 3. Inventory Items, Cost of Goods Sold, and Lead Time (Example)

Inventory Item Name	COGS (Cost of Goods Sold)	*Lead Time* (Days)
Pen	Rp2,950	3
Notebook	Rp6,500	3
Totebag	Rp14,500	3
Module - Mikrotik MTCNA + Exam	Rp51,500	3
T-Shirt - Mikrotik MTCNA + Exam - L, XL, XL, XXL	Rp64,000	3
Module - Cisco CCNA	Rp38,000	3
T-Shirt - Cisco CCNA - L, XL, XL, XXL	Rp63,000	3
Module - ANDROID KOTLIN	Rp30,000	3
T-Shirt - ANDROID KOTLIN - L, XL, XL, XXL	Rp62,500	3
...(and other items)	...	...

Implementation of The Genetic Algorithm

The implementation of the genetic algorithm serves as the core of the optimization system. The process begins with constructing potential solutions, referred to as chromosomes, which are then evaluated and refined through evolutionary procedures. Each chromosome represents a possible inventory solution and consists of paired genes assigned to every item. Specifically, each item is represented by two genes: the Reorder Point (ROP), which defines the minimum stock level as an integer threshold that triggers a reorder when inventory falls below it, and the Reorder Quantity (ROQ), which specifies the number of items to be ordered during replenishment, also expressed as an integer. A chromosome is therefore the aggregation of all these genes arranged in a consistent sequence. Given that the system manages 573 unique inventory items, each chromosome contains a total of 1,146 genes. To illustrate this structure, Python code and an example table are provided, demonstrating how chromosomes are programmatically generated.

Table 4. Example of Chromosome Representation

Inventory Item Name	Gen ROP	Gen ROQ
Pen	15	30
Notebook	10	25
Totebag	20	35
Module - Mikrotik MTCNA + Exam	14	20
T-Shirt - Mikrotik MTCNA + Exam - L	14	20
T-Shirt - Cisco CCNA - XL	8	12

...(and other items)	...	...

```
# Python code to create chromosomes
# (Part of the create_chromosome function)
# Assumptions of parameters and data that have already been loaded
ITEM_NAMES_ORDERED = ['Pen', 'Notebook', 'Totebag'] # Example
ROP_MIN = 0
ROP_MAX = 28
ROQ_MIN = 1
ROQ_MAX = 42

def create_chromosome():
    """
    Creating one new chromosome at random.
    Each chromosome is a list of pairs (ROP, ROQ) for each inventory item.
    """
    chromosome = []
    for item_name in ITEM_NAMES_ORDERED:
        # Taking random ROP and ROQ within a predetermined range
        rop = random.randint (ROP_MIN, ROP_MAX)
        roq = random.randint (ROQ_MIN, ROQ_MAX)
        chromosome.extend([rop, roq])
    return chromosome

# Example of a single chromosome produced
# The result of this code will be one of the rows in the Table
# chromosome = create_chromosome()
# print(chromosome)
# Output will be similar: [15, 30, 10, 25, 20, 35]
```

Fitness Function

The fitness function serves as the "heart" of the genetic algorithm, responsible for evaluating the quality of each chromosome (inventory plan) and assigning a score. Its primary goal is to identify the most efficient solution, defined in this case as a plan that minimizes costs while ensuring product availability. To calculate the fitness score, a simulation is performed for each chromosome over the forecasted demand period of August–September 2025, recording all relevant costs and penalties. The assessment components include purchase costs, calculated from the total expenses of all orders placed during the simulation; holding costs, which are assumed to be zero based on information from PT. ID-Networkers since the warehouse is privately owned with no separate operational expenses; and stockout costs, which act as the most critical metric to guarantee availability. Although no direct financial penalty is incurred, the algorithm assigns an extremely large penalty of Rp 1,000,000,000 per unit of shortage to mathematically enforce the prioritization of availability. The fitness score is computed using the formula Fitness = 1 / (Total Purchase Cost + Total Holding Cost + Total Stockout Cost + 1), which transforms the cost-minimization problem into a fitness-maximization problem. Consequently, the lower the total cost, the higher the fitness value, with a maximum possible score of 1.0. Python code is then used to implement this fitness function along with the underlying simulation logic.

```
# Python code to calculate chromosomal fitness
# (Part of calculate_fitness function)
def calculate_fitness(chromosome):
    # ... (initialization code and simulation loop)
    # ...
    #
    # Inside the simulation loop:
```

```
    # Reduce stock on demand
    # if current_stock[item_name] >= needed:
    #     current_stock[item_name] -= needed
    # else:
    #     stockout_units = needed - current_stock[item_name]
    #     total_stockout_penalty += stockout_units * STOCKOUT_PENALTY_PER_UNIT
    #     current_stock[item_name] = 0
    #
    # Calculating the purchase cost when the order is placed
    # total_purchase_cost += roq * hpp
    # ...
    #
    # After the simulation loop is complete:
    # total_cost = total_purchase_cost + total_holding_cost + total_stockout_penalty
    # fitness = 1 / (total_cost + 1)
    #
    # return fitness, total_purchase_cost, total_stockout_penalty, total_holding_cost
```

Evolutionary Operators

Evolutionary operators are the core mechanisms of the genetic algorithm, responsible for generating new and improved solutions across successive generations. In this system, three operators are applied: selection, crossover, and mutation. The selection process identifies the best-performing chromosomes from the population to serve as parents for the next generation. A tournament selection method is used, in which several chromosomes are chosen at random and the one with the highest fitness value is selected as a parent. To preserve high-quality solutions, an elitism strategy is also implemented, ensuring that the top five chromosomes with the highest fitness scores are automatically retained in the next generation. Once parents are chosen, the crossover operator combines their genes to produce offspring solutions. This is carried out using a one-point crossover method, where chromosomes are split at a random point and the genes beyond that point are exchanged between the two parents. This process allows the algorithm to explore new parameter combinations of ROP and ROQ in search of more optimal solutions. To maintain diversity, a mutation operator is introduced with a low probability (e.g., 5%), randomly altering the value of a gene (ROP or ROQ) in the offspring chromosome. Mutation ensures genetic variation, enabling the algorithm to explore a wider solution space and preventing premature convergence to local optima. Python code is then used to implement these evolutionary operators within the optimization framework.

```
# Python code for evolution operators
# (Part of run_genetic_algorithm function)
def select_parents(population, fitness_scores, elitism_count):
    """
    Selecting parents using Tournament Selection.
    """
    parents = []
    # Maintain elitism (best chromosomes)
    elite_indices = np.argsort(fitness_scores)[::-1][:elitism_count]
    for i in elite_indices:
        parents.append(population[i])

    # Choosing the rest of the parents with tournament selection
    for _ in range(len(population) - elitism_count):
        tournament_size = 5
        tournament_candidates = random.sample(list(zip(population, fitness_scores)),
tournament_size)
        winner = max(tournament_candidates, key=lambda x: x[1])[0]
        parents.append(winner)
    return parents
```

```
def crossover(parent1, parent2):
    """
    Perform a one-point crossover to make two children.
    """
    crossover_point = random.randint(1, len(parent1) - 1)
    child1 = parent1[:crossover_point] + parent2[crossover_point:]
    child2 = parent2[:crossover_point] + parent1[crossover_point:]
    return child1, child2
def mutate(chromosome, mutation_rate):
    """
    Mutations in chromosomes.
    """
    mutated_chromosome = list(chromosome)
    for i in range(len(mutated_chromosome)):
        if random.random() < mutation_rate:
            # ROP or ROQ gene mutations
            if i % 2 == 0: # This is the ROP gene
                mutated_chromosome[i] = random.randint(ROP_MIN, ROP_MAX)
            else: # This is the ROQ gene
                mutated_chromosome[i] = random.randint(ROQ_MIN, ROQ_MAX)
    return mutated_chromosome
```

Main Algorithm Loop

The core process of the genetic algorithm is executed within a loop that runs for a predetermined number of generations (NUM_GENERATIONS). In each iteration, the population of solutions evolves toward increasingly optimal results. The loop begins with the initialization of a population of 100 randomly generated chromosomes, each representing an inventory plan. Every chromosome is then evaluated using the fitness function (*calculate_fitness*), which assigns a score reflecting both cost efficiency and product availability. Based on these fitness scores, the best chromosomes are selected as parents for the next generation. An elitism strategy ensures that the five top-performing chromosomes from the current generation are directly preserved in the new population, while the remainder is selected through Tournament Selection, allowing fitter chromosomes a higher probability of reproduction. The chosen parents then undergo crossover and mutation operations to generate offspring that replace less effective chromosomes in the population. This evolutionary process is repeated for 50 generations, with the algorithm progressively refining its search for optimal solutions. At the end of the loop, the algorithm outputs the best solution discovered, including detailed cost metrics and optimal ROP/ROQ policies, providing concrete recommendations for inventory management at PT. ID-Networkers.

```
# Python code for the main loop of the algorithm
# (Part of run_genetic_algorithm function)

def run_genetic_algorithm():
    # Initial population initialization
    population = create_initial_population(POPULATION_SIZE)

    best_chromosome = None
    best_fitness = 0.0

    for generation in range(NUM_GENERATIONS):
        # Fitness evaluation
        fitness_scores = [calculate_fitness(c)[0] for c in population]

        # Save the best solution
        max_fitness_current_gen = max(fitness_scores)
        if max_fitness_current_gen > best_fitness:
            best_fitness = max_fitness_current_gen
```

```
            best_chromosome = population[fitness_scores.index(max_fitness_current_gen)]

        # Show progress
        print(f"Generation {generation + 1}: Best Fitness = {best_fitness:.10f}")

        # Create new populations through selection, crossovers, and mutations
        new_population = []
        # ... (code for selection, crossover, and mutation)
        population = new_population

    # Show final results
    # ... (code to display the best solution)
```

Final Test Results

This section presents a comprehensive overview of the experimental findings, including a comparison of system performance against the existing manual method and an in-depth analysis of the solutions generated by the genetic algorithm. The testing demonstrated the algorithm's strong optimization capability. After running simulations over 50 generations with a population of 100 chromosomes, the algorithm successfully identified an optimal solution. The convergence analysis revealed that both the highest fitness value (best solution) and the average fitness of the population steadily increased across generations, confirming the algorithm's ability to learn, adapt, and progressively deliver more efficient solutions. In terms of cost efficiency, the algorithm minimized total inventory costs, particularly by avoiding the substantial penalties associated with stockouts. A comparison between the optimal solution's total operational costs (purchase cost) and the worst-case scenario under the manual method—where stockouts frequently occur—showed significant cost savings. Moreover, the simulation using the optimal solution confirmed that no stockouts occurred during the forecast period of August to September 2025, thereby fulfilling one of the algorithm's primary objectives: ensuring product availability for all training activities.

Recommendations for Optimal Inventory Policy

The optimal solution generated by the genetic algorithm provides concrete and implementable inventory policies. Table 4.7 presents a summary of the recommended Reorder Point (ROP) and Reorder Quantity (ROQ) for selected inventory items. For instance, the recommended ROP and ROQ values are 10 and 15 for pens, 12 and 20 for notebooks, and 15 and 25 for tote bags. Similarly, for training-related items, the policy suggests an ROP of 14 and ROQ of 20 for the *Mikrotik MTCNA + Exam* T-shirt (size L), an ROP of 8 and ROQ of 12 for the *Cisco CCNA* T-shirt (size XL), and an ROP of 10 and ROQ of 10 for the *DEVOPS* module. Additional items follow the same framework, ensuring that each unique inventory type is equipped with tailored reorder policies to balance cost efficiency and product availability.

CONCLUSION

This study demonstrates that the application of a Genetic Algorithm significantly improves the optimization of inventory management at PT Integrasi Data Nusantara by simultaneously determining optimal Reorder Points (ROP) and Reorder Quantities (ROQ). The experimental results confirmed that the proposed system successfully eliminated stockout incidents, reduced overall inventory costs, and provided implementable policies that outperform the existing manual method. These findings highlight the algorithm's adaptability in handling dynamic demand fluctuations and its potential to support data-driven, automated decision-making in inventory management. However, the research has certain limitations, particularly in the use of cost assumptions such as zero holding costs and a fixed stockout penalty, as well as the reliance on demand forecasts limited to a two-month period. These constraints may affect the system's generalizability to other operational settings or longer planning horizons. Therefore, future research is recommended to incorporate more comprehensive and realistic cost structures, longer demand forecasting windows, and hybrid optimization techniques that combine Genetic Algorithms with other methods, such as Machine Learning or Particle Swarm Optimization, to enhance accuracy and robustness. By addressing these aspects, subsequent studies can further strengthen the applicability and scalability of optimization systems for complex inventory environments.

REFERENCES

Anasta, N., Kesuma, I., & Wardana, A. (2024). Optimalisasi Sistem Distribusi Tenaga Listrik Menggunakan Algoritma Genetika. *Jurnal Elektronika Listrik Dan Teknologi Informasi Terapan*, *6*(2), 73–77. https://doi.org/10.37338/elti.v6i2.389

Ardi, A. (2025). Model Optimasi Pengelolaan Inventaris Aset Berbasis Web Menggunakan Pendekatan Design Thinking. *Jurnal Ilmiah Manajemen Aset*, *19*, 50–58.

Azaria, D. P. (2025). Penerapan Algoritma Genetika Mutual Selection Untk Penjadwalan Shift Kerja Karyawan Pada PT. Mifa Bersaudara. *Jurnal Multidisiplin Saintek*, *7*(9). https://doi.org/10.8734/Kohesi.v1i2.365

Diasti, A. K., Nugraha, R., Putra, T. E. M., Kusej, I. S., Pratama, R. M., Andaru, A. R., & Septiani, N. W. P. (2025). Aplikasi Inventarisasi Dan Pengelolaan Stok Real-Time Berbasis Mobile dan Web Database. *Seminar Nasional Riset dan Inovasi Teknologi*, *9*(1), 325–331. https://doi.org/10.30998/semnasristek.v9i1.7830

Fajar, M., Azhar, R., Anshori, Y., Laila, R., Rinianty, & Lapatta, N. T. (2025). Optimization of Inventory Management with QR Code Integration and Sequential Search Algorithm: A Case Study in a Regional Revenue Office. *Journal of Applied Informatics and Computing*, *9*(2), 412–420. https://doi.org/10.30871/jaic.v9i2.8919

Fitriyani, A., Lubis, H., Hendharsetiawan, A. A., & Hutahaean, J. A. L. (2025). Sistem Penjadwalan Mata Pelajaran Menggunakan Algoritma Genetika SMK PGRI Rawalumbu. *Jurnal Manajamen Informatika Jayakarta*, *5*(2), 205–219. https://doi.org/10.52362/jmijayakarta.v5i2.1881

Iwasokun, G. B., & Alimi, S. A. (2022). Genetic Algorithm Model for Stock Management and Control. *International Journal of Strategic Decision Sciences*, *13*(1), 1–20. https://doi.org/10.4018/ijsds.309119

Ja'a, W. A. T., Katili, M. R., Wungguli, D., & Yahya, N. I. (2022). Critical Path Method Dan Algoritma Genetika Untuk Optimasi Durasi Dan Biaya Pembangunan. *Euler : Jurnal Ilmiah Matematika, Sains Dan Teknologi*, *10*(2), 292–302. https://doi.org/10.34312/euler.v10i2.14488

Mayyani, H., Nurbaiti, M., Supriyo, P. T., Aman, A., & Silalahi, B. P. (2023). Penerapan Algoritma Genetika Dengan Metode Roulette Wheel Dan Replacement Pada Masalah Memaksimumkan Omzet. *MILANG Journal of Mathematics and Its Applications*, *19*(2), 153–172. https://doi.org/10.29244/milang.19.2.153-172

Mulyo, H. (2018). Penerapan Algoritma Genetika Dalam Efisiensi Persediaan Bahan Baku Mebel Di UD. Mebel Jati. *Jurnal Rekognisi Akuntansi*, *2*(2), 155–165.

Nuradi, F., Tundo, T., Mulyana, D. I., & Lestari, S. (2024). Optimisasi Penjadwalan Kegiatan Guru Pada SMK IDN Boarding School Jonggol Dengan Penerapan Algoritma Genetika. *Jurnal Teknologi dan Rekayasa Sistem Komputer (TEKNOKOM)*, *7*(2), 283–290. https://doi.org/10.31943/teknokom.v7i2.223

Parnadi, M. A., Cahyono, B., & Syafrizal, A. (2025). Aplikasi Mobile untuk Optimalisasi Manajemen Pada Usaha Katering K" Parnadi. *Jurnal Media Akademik*, *3*(6). https://doi.org/10.62281

Rupilele, F. G. J., & Lahallo, F. F. (2024). Optimisasi Pengelolaan Barang Di Universitas Victory Sorong Melalui Perancangan Sistem Inventory Terpadu. *Jurnal Jendela Ilmu*, *5*(1), 30–35. https://doi.org/10.34124/ji.v5i1.183

Sari, A. N., Ilhamsah, H. A., & Mu'alim. (2020). Perencanaan Persediaan Bahan Baku Dengan Metode Economic Order Quantity (EOQ) Menggunakan Algoritma Genetika (AG) (Studi Kasus: PT. XYZ). *Jurnal Teknologi Penerbangan*, *4*(1), 1–10.

Saskia, L., Satrio, I. C., & Widodo, B. (2025). Penerapan Algoritma Genetika Untuk Pencarian Rute Terbaik Antar Jemput Laundry. *Jurnal Informatika Teknologi Dan Sains (JINTEKS)*, *7*(1), 266–271.

Simanjorang, R. M., Simangunsong, A., Arifin, M., & Tampubolon, S. D. (2024). Implementasi Algoritma Genetika Dalam Pengembangan Sistem Pakar Untuk Pemilihan Karier. *Jurnal Media Informatika*, *6*(1), 33–40.

Syam, N., Aksara, L. M. F., & Tajidun, L. M. (2023). Sistem Informasi Penjadwalan Persediaan Barang Menggunakan Algoritma Genetika. *ANIMATOR*, *1*(4), 1–5.