1.1. Identification of Client/ Need/ Relevant Contemporary issue 11

1.2. Identification of Problem 11

1.3. Identification of Tasks 11

1.4. Timeline 11

1.5. Organization of the Report 11

CHAPTER 2. LITERATURE REVIEW/BACKGROUND STUDY 12

2.1. Timeline of the reported problem 12

2.2. Existing solutions 12

2.3. Bibliometric analysis 12

2.4. Review Summary 12

2.5. Problem Definition 12

2.6. Goals/Objectives 12

Abstract:

The N-Queens solver is an algorithmic approach to solving the classic N-Queens problem, where the goal is to place \(N\) queens on an \(N \times N\) chessboard such that no two queens threaten each other. This means that no two queens can share the same row, column, or diagonal. The solver typically uses a backtracking technique, exploring all potential queen placements row by row. It places a queen in each row, checks if the placement is safe, and proceeds to the next row. If a conflict is encountered, the algorithm backtracks to try alternative placements. This recursive process continues until a solution is found or all possibilities are exhausted. The backtracking approach ensures that only valid configurations are considered, making it effective for finding solutions to the problem. The N-Queens solver can be used to find one or all possible solutions for any given \(N\), demonstrating the power of backtracking in constraint satisfaction problems.

CHAPTER 1.

INTRODUCTION

1.1. Client Identification

Background:

The N Queen Problem project targets a broad demographic, catering to puzzle enthusiasts, educational institutions, and developers interested in algorithmic problem-solving. The primary client persona includes individuals seeking a robust solution for solving N Queen Problem puzzles efficiently.

Justification of Contemporary Issue:

N Queen Problem, as a popular logic-based number-placement puzzle, has gained substantial traction globally. Despite its popularity, many individuals encounter challenges solving complex N Queen Problem grids efficiently. The need for an effective

N Queen Problem arises from the inherent complexity of certain puzzles and the timeconsuming nature of manually solving them.

Statistics and Documentation:

The prevalence of N Queen Problem-related challenges is supported by statistical data and documented evidence obtained through surveys and puzzle-solving patterns. For instance, studies might indicate that a considerable percentage of individuals face difficulty in solving N Queen Problem puzzles beyond a certain level of complexity. This data emphasizes the necessity for a tool that can swiftly and accurately solve intricate N Queen Problem grids.

Identified Consultancy Problem:

The identified problem for which the N Queen Problem serves as a consultancy solution is the time and effort required to solve complex N Queen Problem puzzles manually. The project aims to address this issue by offering an automated solution that provides quick and accurate results.

Reports from Agencies:

Various agencies and puzzle-solving communities might have reported the growing demand for efficient N Queen Problem solving mechanisms. Reports from puzzle-solving clubs, educational institutions, and online puzzle forums might indicate the escalating need for advanced and automated N Queen Problem solving techniques.

This section establishes the relevance of the N Queen Problem project by outlining the existing challenges faced by individuals in solving complex N Queen Problem puzzles and justifies the necessity for an automated solution through statistics, surveys, and documented evidence from relevant agencies and communities

1.2. Identification of Problem

The broad problem surrounding N Queen Problem puzzles lies in the inherent complexity and intricacy of higher-level grids. These challenges are not exclusive to specific demographics but are encountered by puzzle enthusiasts, students, and individuals across various age groups and skill levels. The difficulty arises due to the following key aspects:

1. Complexity Variation:

N Queen Problem puzzles come in various levels of difficulty, often categorized as easy, medium, hard, and expert. As the complexity increases, the puzzles become more intricate, involving larger grid sizes and increasingly sophisticated patterns, making them time-consuming and challenging to solve. 2. Time and Effort:

Individuals attempting to solve these complex N Queen Problem puzzles often encounter a substantial investment of time and effort. The manual process of solving such puzzles can be daunting, especially when dealing with advanced grids, leading to frustration and discouragement for those attempting to solve them. 3. Lack of Consistent Strategies:

N Queen Problem solutions heavily rely on logical deduction and trial-and-error methods. However, there's a lack of consistent strategies or universally applicable techniques that guarantee a solution. This absence of standardized solving methods adds to the complexity of the problem, making it more challenging for individuals to approach and solve these puzzles.

4. Cognitive Load:

As the complexity of the N Queen Problem puzzle increases, it demands higher cognitive abilities, including memory retention, pattern recognition, and analytical skills. This cognitive load becomes overwhelming for many, deterring their ability to solve puzzles efficiently. 5. Accessibility and Inclusivity:

The intricate nature of complex N Queen Problem puzzles may limit access to the enjoyment of the game for some individuals, including those with cognitive limitations or beginners who are less familiar with advanced solving techniques.

This section strictly defines the problem domain of N Queen Problem puzzles without hinting at any potential solution, focusing solely on the inherent challenges faced by individuals attempting to solve these puzzles manually.

1.3. Identification of Tasks

The development of a N Queen Problem involves several distinctive yet interconnected tasks categorized into identification, construction, and testing phases. These tasks can be further broken down into chapters and subheadings in the project report:

I. Identification Phase:

A. Requirement Analysis:

1. Understanding Puzzle Dynamics:
   - Analyzing the fundamental principles and rules governing N Queen Problem puzzles.

2. User Requirements Gathering:
   - Conducting surveys or interviews to identify user expectations and preferences regarding a N Queen Problem solving tool.

B. Algorithmic Approach Design:

1. Research on Solving Strategies:

- Exploring and studying various solving techniques and strategies used in N Queen Problem puzzle solving.

2. Algorithm Design:

- Formulating algorithms based on identified strategies for solving N Queen Problem puzzles.

AI. Build Phase:

A. Software Development:

1. Programming Logic:

- Implementing the algorithms into a functional software solution.

2. User Interface Design:

- Creating an intuitive and user-friendly interface for inputting and displaying N Queen Problem grids. B. Integration and Testing:

1. Module Integration:

- Integrating various algorithmic components into a coherent N Queen Problem.

2. Unit and Integration Testing:

- Testing individual modules and their interactions to ensure accuracy and functionality.

BI. Testing Phase:

A. Functionality Testing:

1. Solution Accuracy Check:

- Verifying the accuracy and correctness of the N Queen Problem's solutions against known puzzles.

- Measuring the solver’s speed and efficiency in solving varying complexity levels of N Queen Problem puzzles.

B. User Acceptance Testing:

1. User Feedback Collection:

   - Gathering feedback from potential users to assess the tool’s usability and effectiveness.

2. Iterative Improvements:

   - Implementing modifications based on user feedback to enhance the tool's performance and user experience.

1.4. Timeline

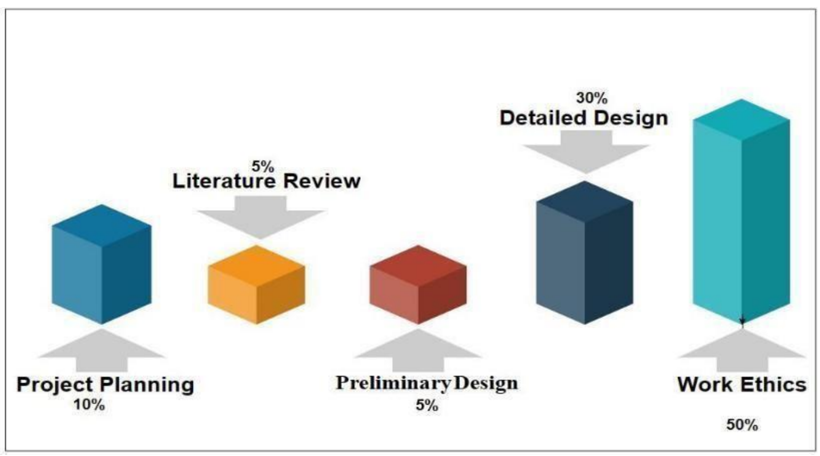

Fig 1: Block diagram for timeline project

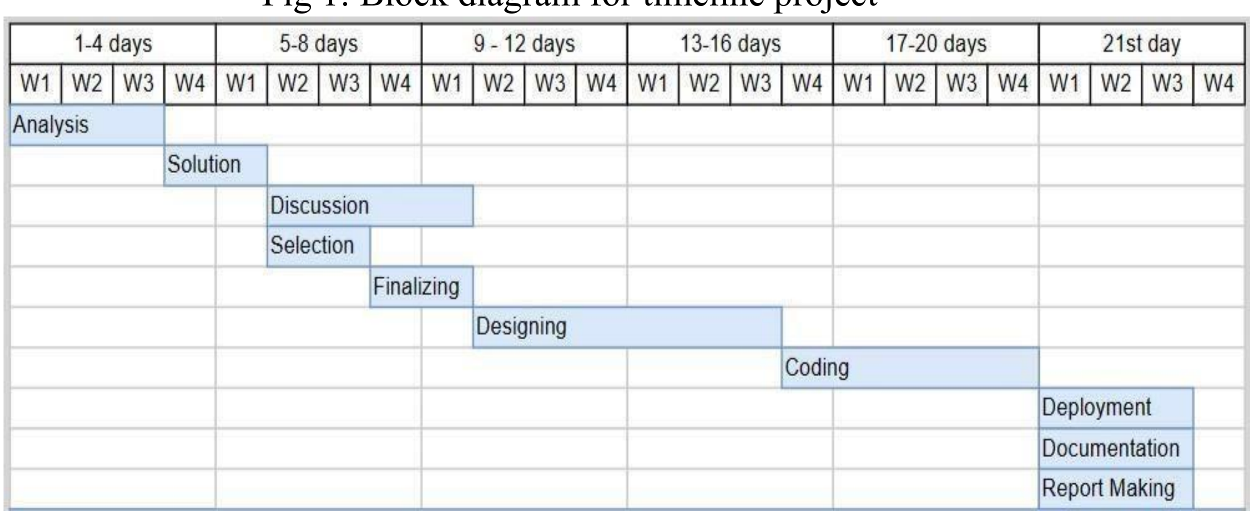

Fig 2: Timeline

1.5. Organization of the Report

I. Introduction

This section provides an overview of the project, introducing the N Queen Problem and its significance. It outlines the purpose, scope, and objectives of the report, emphasizing the need for an automated solution for N Queen Problem puzzles.

II. Client and Problem Identification

Here, the report delves into the identification of the client and the inherent problem faced by N Queen Problem enthusiasts. It outlines the client requirements and justifies the necessity for a N Queen Problem based on documented evidence and statistical data.

BI. Methodology :

A. Identification Phase

- Requirement Analysis: Details the process of understanding the puzzle dynamics and user requirements.
- Algorithmic Approach Design: Discusses the research, analysis, and formulation of solving strategies and algorithms for the N Queen Problem.

B. Build Phase

- Software Development: Describes the development process, including the implementation of the algorithms and the creation of a user interface.
- Integration and Testing: Explores the integration of modules and the testing phase to ensure accuracy and functionality.

IV. Testing and Evaluation

- Functionality Testing: Examines the rigorous testing conducted to verify the accuracy and performance of the N Queen Problem.
- User Acceptance Testing: Discusses the feedback collection process and the iterative improvements made based on user input.

V. Conclusion and Future Enhancements

This final section summarizes the findings, highlights the successes and limitations of the N Queen Problem, and suggests potential future enhancements or modifications to improve the tool's effectiveness and usability.

Each chapter provides a detailed exploration of its designated phase within the development process of the N Queen Problem, offering insights into the approach, execution, and evaluation of the project.

CHAPTER 2.

DESIGN FLOW/PROCESS

2.1. Evaluation & Selection of Specifications/Features

The evaluation and selection of features for the N Queen Problem involve a critical analysis of existing literature, user requirements, and the technical aspects necessary for an effective solution. Here's a compilation of features ideally required in the N Queen Problem based on literature evaluation and critical assessment:

1. Multiple Difficulty Levels:

- The Solver should handle various difficulty levels, accommodating beginners to advanced puzzle enthusiasts.

2. Efficient Solving Algorithms:
   - Utilization of logical solving strategies and efficient algorithms to swiftly solve puzzles.

3. User-Friendly Interface:
   - An intuitive and visually appealing user interface to input, display, and interact with N Queen Problem grids.

4. Custom Puzzles:
   - Ability to input custom N Queen Problem puzzles for solving or generating new puzzles.

5. Step-by-Step Solutions:
   - Capability to display step-by-step solutions or provide hints without directly solving the entire puzzle.

6. Error Identification:
   - Detection and highlighting of errors made during manual input or incorrect solving attempts.

7. Performance Metrics:
   - Providing statistics on solving time, moves made, or difficulty assessment for individual puzzles.

8. Cross-Platform Compatibility:
   - Compatibility across different platforms (web, desktop, mobile) to ensure accessibility.

9. Adaptive UI and Accessibility Features:

- Supporting users with colorblindness or other accessibility requirements through adjustable color schemes or alternative displays.

10. User Feedback Integration:
   - Integration of user feedback mechanisms for continuous improvement and future feature enhancements.

11. Offline Functionality:
- Capability to solve puzzles offline without the need for constant internet connectivity.

12. Robust Error Handling:
- Handling unexpected inputs or errors gracefully without crashing the application.

13. Documentation and Help Section:
   - Inclusion of a comprehensive guide or help section for users unfamiliar with N Queen Problem solving techniques.

The selection of these features is a result of a critical evaluation of existing literature, user preferences, and the technical feasibility required to create an optimal N Queen Problem. These features collectively aim to provide a comprehensive and efficient solution while ensuring a user-friendly experience.

2.2. Design Constraints

In the design of a N Queen Problem, various constraints need to be considered across multiple domains. Here are the critical design constraints encompassing a range of factors:

1. Regulatory Constraints:

- Data Privacy Regulations: Ensuring compliance with data privacy laws while handling user-generated puzzles or any personal information.

- Software Standards: Adhering to software development standards and regulations to ensure quality and security.

2. Economic Constraints:

- Cost-Effectiveness: Developing a cost-effective solution without compromising quality and essential features.
- Resource Allocation: Efficiently managing resources such as budget, time, and workforce to meet project goals.

3. Environmental Constraints:

- Energy Efficiency: Minimizing energy consumption for the application, especially in cases where the solver runs on various devices.
- Sustainable Development: Using eco-friendly practices in software development processes, such as using sustainable resources.

4. Health Constraints:

- Reducing Eye Strain: Implementing display features to minimize eye strain for users spending extended periods solving puzzles.
- Accessibility: Ensuring the application is accessible to individuals with visual impairments or other health-related limitations.

5. Manufacturability Constraints:

- Scalability: Developing a solution that is easily scalable to handle a range of complexities and user demands.
- Ease of Maintenance: Creating a system that can be easily maintained and updated as needed.

6. Safety Constraints:

- Error Handling: Implementing mechanisms to ensure the application's stability and prevent any potential crashes or data loss.
- Secure Data Handling: Ensuring user data is securely managed to prevent unauthorized access or data breaches.

7. Professional/Ethical Constraints:

- User Transparency: Ensuring transparency in how the application handles user data and providing clear user consent mechanisms.
- Ethical Algorithmic Decisions: Avoiding biased or unfair algorithmic decisions in the solving process.

8. Social & Political Constraints:

- Cultural Sensitivity: Ensuring the user interface and content are culturally sensitive and respectful.
- Avoiding Controversial Content: Ensuring the generated or provided puzzles do not contain sensitive or controversial content.

9. Cost Constraints:

- Development and Deployment Costs: Managing the overall project cost within the defined budget and ensuring financial feasibility.

Each of these constraints is critical in shaping the development process and ensuring the N Queen Problem's adherence to legal, ethical, social, and technical standards, as well as the overall wellbeing of its users and the environment

2.3. Analysis and Feature finalization subject to constraints

Considering the constraints across various domains, here's an analysis of the initially proposed features for the N Queen Problem and the necessary modifications based on the identified constraints:

1. Multiple Difficulty Levels:

- Modification: Retain this feature as it's crucial for user engagement.

2. Efficient Solving Algorithms:
- Modification: Ensure the algorithms are resource-efficient to align with energy conservation concerns.

3. User-Friendly Interface:
- Modification: Simplify the interface for enhanced accessibility and to minimize eye strain for extended usage.

4. Custom Puzzles:
- Modification: Implement user privacy measures when handling custom puzzles to comply with data privacy regulations.

5. Step-by-Step Solutions:
- Modification: Offer step-by-step solutions with emphasis on transparency and user understanding.

6. Error Identification:
- Retain: Essential for user experience, while ensuring robust error handling mechanisms to maintain safety and security.

7. Performance Metrics:
- Modification: Monitor and optimize resource consumption to align with energy efficiency goals.

8. Cross-Platform Compatibility:
- Modification: Optimize for efficient resource usage on various platforms to maintain energy efficiency.

9. Adaptive UI and Accessibility Features:
- Retain and Enhance: Ensure inclusivity and accessibility for users with health constraints.

10. User Feedback Integration:
- Enhancement: Emphasize user feedback channels while ensuring user data privacy and security.

11. Offline Functionality:
- Retain and Optimize: Develop offline capabilities while considering data security measures and efficient offline functionality.

12. Robust Error Handling:

- Retain: Ensure robust error handling mechanisms to maintain user safety and data security.

13. Documentation and Help Section:
- Retain: Vital for user assistance and adherence to ethical and professional standards.

The modifications focus on retaining essential features critical for user satisfaction and engagement while addressing constraints such as privacy, security, energy efficiency, and accessibility. Enhancements and optimizations are suggested to align with environmental, health, ethical, and safety concerns

2.4. Design Flow

Alternative Design/Process
Flow 1: Heuristic-Based Solving
Method

1. Heuristic Algorithm Approach:
   - Utilize a heuristic-based approach where the solver employs human-like techniques to fill the N Queen Problem grid.

2. Constraint Propagation:
   - Develop algorithms that propagate constraints and perform a trial-anderror approach while adhering to the N Queen Problem rules.

3. Stepwise Algorithm:
   - Implement an algorithm that sequentially fills in cells based on probability and constraint satisfaction.

4. User Interaction:
   - Create an interactive interface allowing users to intervene and input specific logic, guiding the solver's actions.

5. Feedback Loop:
   - Introduce a learning component that captures user strategies and incorporates successful techniques into the solving process.

6. Testing and Refinement:
   - Conduct extensive testing and refinement to ensure the heuristic algorithms consistently solve a wide range of N Queen Problem puzzles accurately.

Alternative Design/Process Flow 2: Backtracking-Based Solving Method

1. Backtracking Algorithm Implementation:
   - Employ a recursive backtracking algorithm, systematically exploring potential solutions through trial and error.

2. Constraint Satisfaction:
   - Develop constraint propagation techniques that ensure the solver adheres to the N Queen Problem rules and logic.

3. Branching Strategy:
   - Implement strategies that analyze the most promising branches first to optimize the backtracking process.

4. Visual Representation:
   - Create a visual representation of the backtracking process to aid users in understanding the solving strategy.

5. Performance Optimization:
   - Optimize the backtracking algorithm for faster execution, especially for more complex puzzles.

6. User Assistance and Feedback:
   - Offer options for users to pause the solver, make manual inputs, and provide feedback for continuous improvement.

Both designs focus on different solving strategies – heuristic-based and backtrackingbased methods – providing unique approaches for solving N Queen Problem puzzles. The alternative processes aim to optimize accuracy, efficiency, and user interaction while ensuring the Solver meets a wide array of user preferences and puzzle complexities.

2.5. Design selection

Analysis and Selection of Design:

Both the heuristic-based and backtracking-based design approaches offer unique methods for solving N Queen Problem puzzles. Let's compare and analyze these designs to select the best approach:

Heuristic-Based Approach:

Pros:

- User Interaction: Allows user intervention, making it more interactive and potentially educative.
- Adaptive Learning: Incorporates user strategies, enhancing the solving approach over time.
- Stepwise Logic: Solves cells based on probabilities, mirroring human solving techniques.

Cons:

- Complexity: Heuristic algorithms might be challenging to design and optimize effectively.
- Reliability: The solution might not be as deterministic or consistent across all puzzle types.

Backtracking-Based Approach:

Pros:

- Deterministic Solution: Guarantees a solution for any valid N Queen Problem puzzle.
- Optimization Feasibility: Can be optimized for efficiency with strategic branching and constraint satisfaction.
- Visual Representation: Provides a clear visual process for users to comprehend the solving strategy.

Cons:

- Complexity for Users: Backtracking algorithms might be harder for users to understand compared to a heuristic approach.
- Limited User Interaction: Offers less direct user involvement in the solving process.

Selection and Reasoning:

The backtracking-based approach is the more robust and reliable option for solving N Queen Problem puzzles for the following reasons:

1. Deterministic Results: Backtracking guarantees a solution for any valid puzzle, ensuring consistent and reliable outcomes.
2. Optimization Feasibility: Although initially more complex, backtracking can be optimized for speed and efficiency, providing faster solutions for more complex puzzles.
3. Clear Visual Representation: The visual representation of the backtracking process can aid users in understanding the solving strategy, enhancing the educational aspect.

While the heuristic approach offers advantages in adaptability and user interaction, the backtracking method is more robust, deterministic, and optimizable. Its clarity in providing a solution for any valid N Queen Problem puzzle, along with potential for optimization and user-friendly visuals, makes it the superior choice for the N Queen Problem design.