QUESTION PAPER GENERATOR SYSTEM USING ANT COLONY OPTIMIZATION ALGORITHM

Team Details

Roll No.	Name of Student	Email Id	Mobile No.
CT21098	Ambika Srivastava	srivastava29ambika@gmail.com	8408874623
CT21004	Aachal Dhenge	aachaldhenge@gmail.com	8767920006
CT21058	Omkar Khatik	omkarkhatik07@gmail.com	9860325655
CT21018	Sanskruti Lanjewar	lanjewarsanskruti@gmail.com	8999151952
CT21060	Nishad Raut	nishadraut@gmail.com	7499698921

Abstract

This project presents the development of an Automatic Question Paper Generator utilizing the Ant Colony Optimization (ACO) algorithm. The system addresses the challenges of manual question paper generation by automating the process and ensuring the generated paper meets specified criteria such as difficulty level, branch, semester, subject, and module. The ACO algorithm, inspired by the foraging behavior of ants, helps in selecting questions that are most suitable, balancing the paper according to the required difficulty and topic coverage. Pheromone levels and heuristic values like question difficulty play a critical role in the iterative process, where the algorithm continuously refines the selection to improve the quality of the paper.

The system offers a user-friendly interface that allows admins to input, update, and manage a vast pool of questions stored in a MySQL database. The ACO algorithm is designed to create multiple possible question sets in each iteration, evaluate them, and update pheromone trails based on the quality of the generated solutions. Over multiple iterations, the system converges to the most optimal question paper that satisfies the user-defined parameters. Built with Java and Spring Boot for the backend and MySQL for the database, this system provides a scalable and efficient solution for educational institutions. It reduces manual workload, minimizes bias, and ensures consistency and fairness in question paper generation. Furthermore, the system is highly customizable, making it adaptable for different academic requirements across various courses and programs. This project has the potential to enhance examination processes by offering an intelligent, automated approach to question paper creation, ultimately improving both the efficiency and quality of academic assessments.

Aim

The Aim of “Developing Automatic Question paper Generation“ is to design and develop an automated system for generating question papers that ensures a balanced distribution of questions based on predefined criteria such as difficulty level, topic coverage, and diversity

Objectives

- To make the question paper generation process faster.
- To implement the Ant Colony Optimization (ACO) algorithm for intelligently selecting and optimizing questions from a large database.
- To save time and effort of teachers.
- To provide error free and reliable management system.

Introduction

The Automatic Question Paper Generator System using Ant Colony Algorithm is designed to automate the process of generating question papers by using the Ant Colony Optimization algorithm, a powerful optimization technique inspired by the behaviour of ants. This project addresses the limitations of traditional manual question paper creation methods, which are often time-consuming and prone to bias, by providing an intelligent and adaptive system. The system generates question papers based on predefined criteria, such as difficulty level, subject, branch, and module, by selecting questions from a large database. The Ant Colony Optimization ACO algorithm helps the system optimize the selection of questions by simulating the pheromone trails left by ants in nature. As ants (or in this case, algorithms) explore potential solutions, they update pheromone levels, guiding future iterations toward more optimal solutions that meet the requirements. The proposed architecture includes key modules such as a login system for admin and users, a question addition module for the admin, and a question paper generation module that takes user inputs like difficulty level and subject. Over time, the algorithm learns from past iterations to generate better question papers that match the desired criteria. The system is scalable and adaptable, making it a practical solution for educational institutions. This project report outlines the architecture, implementation, and algorithmic details of the system, demonstrating the effectiveness of ACO in optimizing the generation process. It offers a reliable and efficient way to

automate question paper creation, reducing manual effort while ensuring a balanced and requirement-based selection of questions.

Literature Review

- **Automatic question paper generator system by keyword based shuffling algorithm**

Automatic test paper generation selects questions from a database to create a test paper, relying on a large question pool, advanced algorithms, and careful consideration of question quality and relevance to ensure effective assessments. Previously the examination cell of the college or board needed to prepare question papers manually which was very monotonous and time-consuming. There are a few systems in today's market that offer similar services to the proposed system. Those systems are developed by different developers with different features. In this system, we have strived to overcome the drawbacks presented in these systems such as non-portability, static databases, one-tier specifications and much more. Such as: Specific questions as per constraints given by the admin with nothing extra added in. No repetition of questions in the paper.

- **Question Paper Generator System.**

This paper describes the utilization of randomization algorithm in an Automatic Question paper Generator System which has been implemented specially for autonomous institutes. The endeavour needed for generating question paper is diminished after the implementation of this advanced system and because of this advanced system there is no obligation for humans to ponder and employ time which can be utilized on some additional important duty instead of designing question paper. This paper explains the generation of question papers through randomization technique. This technique is used in various schools, institutes, and universities. While generating a question paper the setter has to huge issue of question selection and checking its difficulty level. This paper explains ways to avoid this issue. This paper targets on restricted access to users.

- **Automatic Question Paper Generator System**

In this paper, a fuzzy logic based model is developed for autonomous paper generation, using Python. The software can be used effectively to generate question papers based on the level of the examination which includes unit tests. The question paper then may be referred to a higher authority that has the final decision in these

matters restriction of paper-based systems as majority of human working method, this system agonizes due to bias. There might be few questions duplicated in most of the papers as the professor has a personal tendency towards them. So, there is no assurance of virtuous randomly generated question paper.

Proposed Approach and System Architecture

The extended architecture will follow a modular and scalable approach, ensuring that the new features integrate seamlessly with the existing system. Below is the high-level architecture:

1. Frontend (Client-Side):
    - The UI/UX will include additional pages and forms for:
    - Student login and access to mock MCQ tests.
    - Topic-specific paper generation for educators.
    - Options for PDF customization (e.g., fonts, layout, and watermarking).
    - MCQ paper generation settings.
2. Backend (Server-Side):
    - Use Spring Boot to extend existing RESTful APIs.
    - Implement new modules for:
    - Mock MCQ test generation and evaluation.
    - Topic-specific paper generation logic.
    - PDF customization features.
    - MCQ-specific paper selection using the ACO algorithm.
3. Database (MySQL):
    - Add new tables and fields to store:
    - Mock test results for students.
    - Metadata for topic-based questions.
    - PDF customization preferences.
    - MCQ-specific questions and configurations.
4. Workflow with Ant Colony Optimization:
    - Extend the ACO algorithm to support MCQ-based question selection and topic filtering.
    - Incorporate additional constraints (e.g., include only MCQs or questions from specific topics).

Technologies Used:

Frontend: HTML/CSS/JavaScript.

Backend: Java with Spring Boot.

Database: MySQL for question storage.

Proposed System Flow:

Student Mock Test Flow:

- Login → Select Mock Test → Fetch Questions → Submit Answers → Evaluate Results → Store Results.

Educator Topic-Based Paper Generation Flow:

- Login → Select Topic(s) → Generate Paper → Customize PDF → Download/Save.

PDF Customization Flow:

- Select Paper → Customize Preferences (Fonts, Layout, etc.) → Generate Preview → Confirm Download.

MCQ Paper Generation Flow:

- Login → Select MCQ-only Option → Generate Paper → Download as PDF.

System Architecture

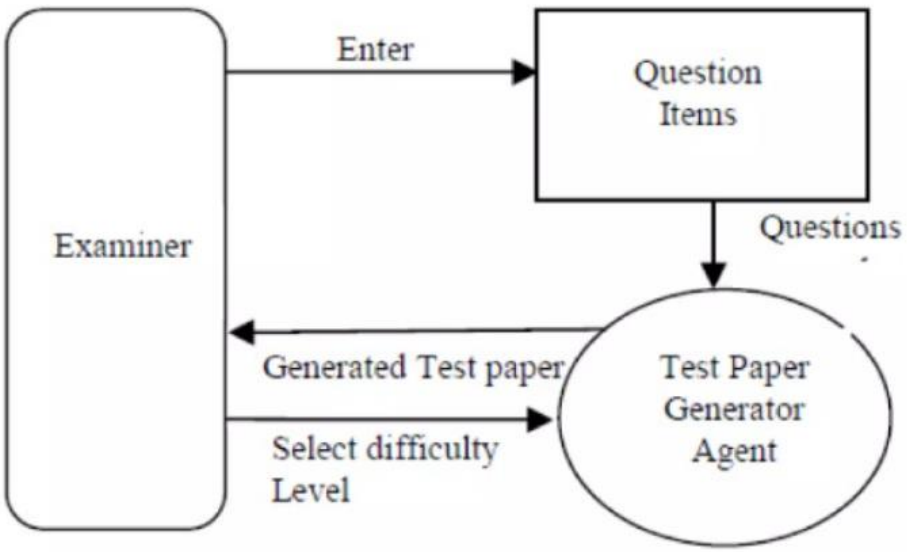

Fig: System Architecture of Automatic Question Paper generator.

Plan of Implementation

The project implementation begins with gathering and analysing requirements to define the scope and technical needs. The system uses architectural diagrams and a database schema to manage users and questions. The development includes modules for user management, question management, and question paper generation using the Ant Colony Optimization algorithm. The front end is built with HTML, CSS, and JavaScript, while the back end is developed using Java and Spring Boot, with MySQL as the database. After integration, the system undergoes thorough testing to ensure functionality and performance. The application is deployed on a server or cloud platform and supported with documentation and user training. Feedback is gathered for ongoing evaluation and maintenance to improve the system.

Tools and Technologies

Programming Languages

- Java: Core language for backend development.
- HTML, CSS, JavaScript: For designing and developing the frontend interface.

Frameworks and Libraries:

- Spring Boot: For building and managing the backend, REST APIs, and business logic.

Database Management System:

- MySQL: For storing and managing data, including questions, user details, and generated question papers.

Algorithm:

- Ant Colony Optimization (ACO): Used to dynamically generate question papers based on input parameters like difficulty level, subject, and module.

Tools:

- IntelliJ IDEA/Eclipse: For coding and backend development.
- VS Code: For front-end development.
- Postman: For testing and validating REST APIs.
- Git/GitHub: For version control and collaboration.

Deployment and Hosting:

- Apache Tomcat/Cloud Server: For hosting the application.

These tools and technologies work together to deliver an efficient, user-friendly, and robust Automatic Question Paper Generator system.

References

1. Mr. Aditya Sanjay Shendure, Mr. Aniruddh Dagadu Khamkar, Mr. Prasanna Premnath Vathare, Mr. Avadhut Salavi (Kumbhar), (2022), “Question Paper Generator System”.
2. Fenil Kiran Gangar, (2018).” Automatic Question Paper Generator System”.
3. Gauri Nalawade, (2017).” Automatic Generation of Question Paper from User Entered Specifications using a Semantically Tagged Question Repository”.
4. Rohan Bhirangi, Smita Bhoir, (2016).” Automated Question Paper Generation System”.
5. Dan Liu, Jianmin Wang, Lijuan Zheng, (2013), “Automatic Test Paper Generation”.
6. Kapil Naik, Shreyas Sule, Shruti Jadhav, Surya Pandey,” Automatic Question Paper Generation System using Randomization Algorithm”.