**CHETTINAD COLLEGE OF ENGINEERING AND TECHNOLOGY, KARUR**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**MINI PROJECT REPORT**

**FINAL EVALUATION**

**Class:** III-CSE/ IV Semester

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Batch No** | **Batch Members** | **Guide Name** | **Title of the Project** | **Accepted/Not Accepted** |
| 09 | Janaki S  Nasika K  Sarmila B | Mrs.C.Sangeetha | Animecode | Accepted |

**ABSTRACT:**

AnimeCode is a web-based platform designed to simplify the learning and teaching of algorithms through an intuitive interface, multi-language code support, visualizations, quizzes, and video recommendations. It enables users to explore algorithms in Python, C, C++, and Java, promoting a better understanding of algorithmic concepts. By integrating various tools and resources, AnimeCode caters to the needs of students, developers, and educators, making algorithm learning engaging and accessible.

# INTRODUCTION

Algorithms are foundational to computer science and software development, but many learners face challenges in understanding and implementing them. AnimeCode addresses these challenges by providing an interactive and centralized platform. It combines algorithm visualizations, multi-language implementations, quizzes, and video tutorials to offer a holistic learning experience. The platform is particularly beneficial for beginners and educators, bridging the gap between theory and practical application.

# OBJECTIVE

* Simplifying the learning of algorithms through engaging visuals.
* Supporting multiple programming languages to cater to diverse learners.
* Providing interactive quizzes to test and reinforce understanding.
* Enabling access to curated video tutorials for deeper insights.
* Offering a user-friendly interface for students, educators, and developers.

# COMPONENTS/LANGUAGES USED

Frontend:

HTML/CSS: For structuring and styling the user interface.

JavaScript: For adding interactivity, animations, and quizzes.

Backend:

PHP: To handle user authentication, data processing, and server-side logic.

Database:

MySQL: For storing user data, quiz results, and algorithm content.

APIs:

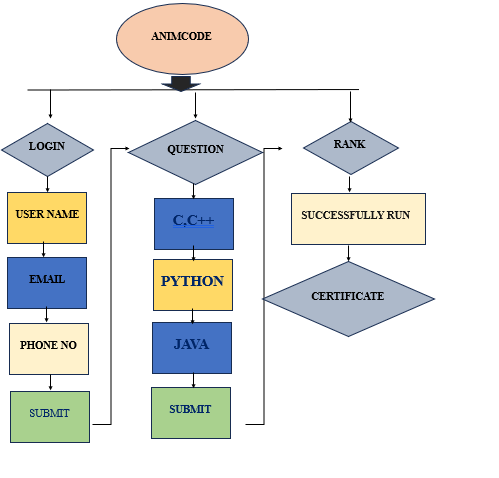
YouTube Data API: For fetching educational video recommendations.

Libraries/Tools:

P5.js: For visualizing algorithm steps.

Firebase Authentication: For secure and scalable user authentication.

**BLOCK DIAGRAM /FLOWCHART**



# METHODOLOGY

Development Steps:

Requirement Analysis: Understand user needs and define platform features.

Frontend Development: Create a responsive interface using HTML, CSS, and JavaScript.

Backend Setup: Implement database connections, user authentication, and content management using PHP and MySQL.

Visualization: Use P5.js to provide algorithm animations.

API Integration: Incorporate the YouTube Data API for video recommendations.

Testing: Conduct usability and functionality testing for seamless user experience.

Deployment: Host the platform on a web server for public access.

**APPLICATIONS**

Education: Aiding students and teachers in understanding algorithms interactively.

Skill Development: Assisting developers in mastering algorithms for interviews.

Self-Paced Learning: Providing learners with quizzes and resources to track progress.

Teaching Support: Facilitating educators with algorithm visualizations and multi-language support.

**REFERENCES**

Python, C, C++, and Java official documentation.

Firebase and YouTube Data API documentation.

Algorithm visualization and educational tools.

"Introduction to Algorithms" by Cormen, Leiserson, Rivest, and Stein.

**WORK PROGRESS REPORT (By students)**

|  |
| --- |
|  |

**WORK PROGRESS REPORT (By Guide)**

|  |
| --- |
|  |

**OVERALL PERFORMANCE OF THE BATCH (By Guide)**

|  |
| --- |
|  |

**Guide Signature Class In-charge HoD/CSE**