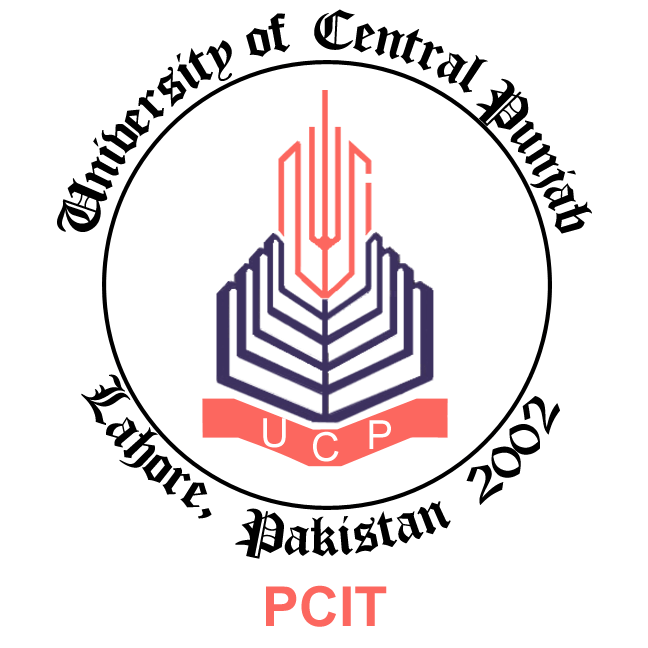
**BSCS FINAL PROJECT PROPOSAL**

CodeFlow

*Term of Registration: Fall 2023*



Presented by:

|  |  |
| --- | --- |
| **Registration No:** | **Name:** |
| L1F20BSSE0191 | MUHAMMAD HASEEB NAWAZ |
| L1F20BSSE0183 | MUHAMMAD MUJEEB |
| L1F19BSSE0073 | ROHAN QAMAR |

|  |
| --- |
| Faculty of Information Technology |

University of Central Punjab

**CodeFlow**

CodeFlow: Visual Programming Tool

**Project Advisor**

Mohsin Sami

**Particulars of the students:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S.No** | **Registration#**  eg.**L1F00BSCS0101** | **Name in Full**  Use Block Letters | **CGPA** | **Signatures** |
| 1 | **L1F20BSSE0191** | **MUHAMMAD HASEEB NAWAZ** | 2.5 |  |
| 2 | **L1F20BSSE0183** | **MUHAMMAD MUJEEB** | 2.25 |  |
| 3 | **L1F19BSSE0073** | **ROHAN QAMAR** | 2.43 |  |

**Advisor’s Consent**

I Prof./Dr./Mr./Ms. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ am willing to guide these students in all phases of above-mentioned project as advisor. I have carefully seen the Title and description of the project and believe that it is of an appropriate difficulty level for the number of students named above.

|  |  |  |
| --- | --- | --- |
| **Note:**  Advisor can’t be changed without prior permission of the Manager Projects and the duration for completion of the Project is 2 regular semesters (approx.) from the date of Registration of Research Project. | Signatures and Date  |  | | --- | |  |   **Advisor** |

**EVALUATOR/REFEREE 1**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| I have carefully read the project proposal and feel that the proposed project is a useful one and of a sufficient difficulty level to justify 2 regular semesters workload for above mentioned students. I have made recommendations in the evaluation form to improve the scope and quality of the project. | | | | | |
|  | | | | Signatures and Date | |
|  |  |  |  |  |  |
|  | | | |  |

**EVALUATOR/REFEREE 2**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| I have carefully read the project proposal and feel that the proposed project is a useful one and of a sufficient difficulty level to justify 2 regular semesters workload for above mentioned students. I have made recommendations in the evaluation form to improve the scope and quality of the project. | | | | | |
|  | | | | Signatures and Date | |
|  |  |  |  |  |  |
|  | | | |  |

**Abstract / Executive Summary**

The CodeFlow project is an innovative web-based solution designed to address the challenges faced by novice programmers and students in understanding programming concepts. It offers an intuitive drag-and-drop interface for creating program logic using flowchart components that is converted into code, a sophisticated code-to-flowchart conversion algorithm, step-by-step visual execution and memory map visualization during program execution. By bridging the gap between abstract code and visual representation, CodeFlow aims to make programming education more accessible, comprehensible, and engaging for learners..

**Introduction and Background**

The field of programming presents significant opportunities, but novice learners often struggle to grasp the abstract nature of code and the complexities of constructing logical program flows. Traditional text-based programming can be intimidating and discouraging for beginners. CodeFlow was conceptualized as a solution to these challenges. It leverages web-based technology and interactive visual elements to create a more approachable learning environment..

**Statement of the Problem**

Novice programmers and students often find it challenging to understand programming concepts due to the abstract nature of textual code and the difficulties in creating logical program flows. This lack of comprehension can hinder their progress and dampen their enthusiasm for programming. Existing solutions, while beneficial have limitations, such as not directly translating to traditional programming languages, does not offer to convert existing code to a visual representation and memory map visualization. CodeFlow aims to overcome these obstacles by providing an intuitive interface and accurate code-to-flowchart conversion.

**Objective(s) / Aim(s) / Target(s)**

The primary objective of CodeFlow is to empower novice programmers and students by making programming concepts more accessible and comprehensible.

Specific aims include:

* Providing an intuitive drag-and-drop interface for code construction.
* Offering an accurate code-to-flowchart conversion algorithm.
* Step-by-Step visual execution of program.
* Incorporating memory map visualization during program execution.
* Fostering creativity, engagement, and motivation among learners.

**Completeness Criteria**

* The platform provides a user-friendly drag-and-drop interface that convert flowchart to code.
* The code-to-flowchart conversion algorithm accurately translates textual code into visual flowcharts.
* Step-by-step visual execution of the program/flowchart.
* Memory map visualization during program execution.

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Criteria** | **Weightage %** |
| 1 | Web Application | 10 |
| 2 | Drag-&-Drop Flowchart feature | 10 |
| 3 | Flowchart to code  Algorithm Development | 20 |
| 4 | Code to flowchart  Algorithm Development | 20 |
| 5 | Step-by-step  visual execution | 20 |
| 6 | Memory map visualization | 15 |
| 7 | Communication | 5 |

**Challenges**

Developing CodeFlow will require overcoming several technical challenges. At the core of the platform's functionality lies the precise conversion of visual flowchart into code and code into flowchart, One of the foremost hurdles lies in accurately mapping flowchart elements to corresponding code segments, especially when dealing with complex control structures like loops and conditionals is a task that presents considerable complexity. Crafting an algorithm capable of performing these conversion accurately demands careful consideration and innovative problem-solving. Simultaneously, the creation of an intuitive drag-and-drop interface, tailored for novice learners, poses its own set of challenges, especially when dealing with the intricacies of programming logic. Furthermore, Step-by-Step visual execution require to ensure real-time synchronization between the visual representation, code execution and CodeFlow's memory map visualization require efficiently managing memory visualization while the program runs step by step requires careful memory management and synchronization.

**Knowledge Areas Required**

* Programming languages, especially C++, will be instrumental in developing the core functionality of CodeFlow, including the conversion of flowcharts into code and vice versa.
* Understanding algorithms and data structures will be crucial for implementing the code-to-flowchart and flowchart-to-code conversion algorithms efficiently and accurately.
* Knowledge of user interface design principles and usability will be essential for creating an intuitive drag-and-drop interface
* Understanding computer architecture is require for memory map visualization and efficient memory management.
* Knowledge of databases are required to store data in a database.

**Learning Outcomes**

In this project, we'll embark on a learning journey that involves several new technologies. We'll delve into the world of web development by creating user interfaces using React.js, a widely-used JavaScript library. Our frontend stack will encompass JSX, CSS, and JavaScript. On the backend, we'll explore Node.js for server-side operations, and we'll incorporate C++ for specific tasks like conversion algorithms, all while managing data using MySQL as our database solution. Additionally, we'll construct APIs to facilitate seamless communication between the frontend and backend components of our project.

**Nature of the End Product / Research Outcomes**

The end product of CodeFlow will be a web-based educational platform tailored for novice programmers and students. This intuitive and interactive application will empower users to grasp programming concepts with ease. Its user-friendly interface, featuring drag-and-drop functionality, will simplify the creation of program logic that can convert your flowchart into textual code. CodeFlow will offer code visualization capabilities, allowing users to convert textual code into visual flowcharts, providing a tangible representation of program logic. Furthermore, the platform will enable learners to observe data storage and manipulation during program execution through memory map visualization, facilitating a deeper understanding of memory management concepts.

**Related Work / Literature Survey / Literature Review**

**Raptor:**

Raptor is a flowchart-based programming software designed to simplify coding for beginners. It shares similarities with your project in terms of providing a visual representation of code logic. Raptor's primary approach is based on creating flowcharts, which is beneficial for visualizing program logic. However, this approach may not directly translate to traditional programming languages that predominantly use textual code.

**Scratch:**

Scratch is a visual programming language developed by the MIT Media Lab. It provides a graphical interface where users can drag and drop code blocks to create interactive stories, games, and animations. Scratch is entirely block-based, making it highly accessible for beginners. CodeFlow, on the other hand, incorporates both textual and visual elements, allowing users to gradually transition to text-based programming languages. Scratch is often used for creating interactive animations and simple games. CodeFlow's focus extends to more comprehensive programming concepts, including memory management, to support a deeper understanding of programming principles.

**Deliverables / Work Breakdown Structure**

**1. Drag and Drop Interface to Make Flowchart:**

**Frontend Development:**

* Develop the drag-and-drop user interface.
* Implement components for adding and connecting flowchart elements.
* Create a user-friendly design and layout.

**2. Convert the Flowchart into Textual Code:**

**Backend Development:**

* Design and develop algorithms to convert flowcharts into textual code.
* Implement a code generation system.
* Ensure the accuracy of the code conversion.

**3. Convert Textual Code into Flowchart:**

**Backend Development:**

* Design and develop algorithms to convert textual code into flowcharts.
* Implement a code-to-flowchart conversion system.
* Validate and refine the visual representation of code.

**4. Step-by-Step Flowchart Visual Execution:**

**Frontend Development:**

* Create an interface for users to execute flowcharts step by step.
* Implement controls for pausing, resuming, and navigating through the execution.
* Highlight the currently executing elements in the flowchart.

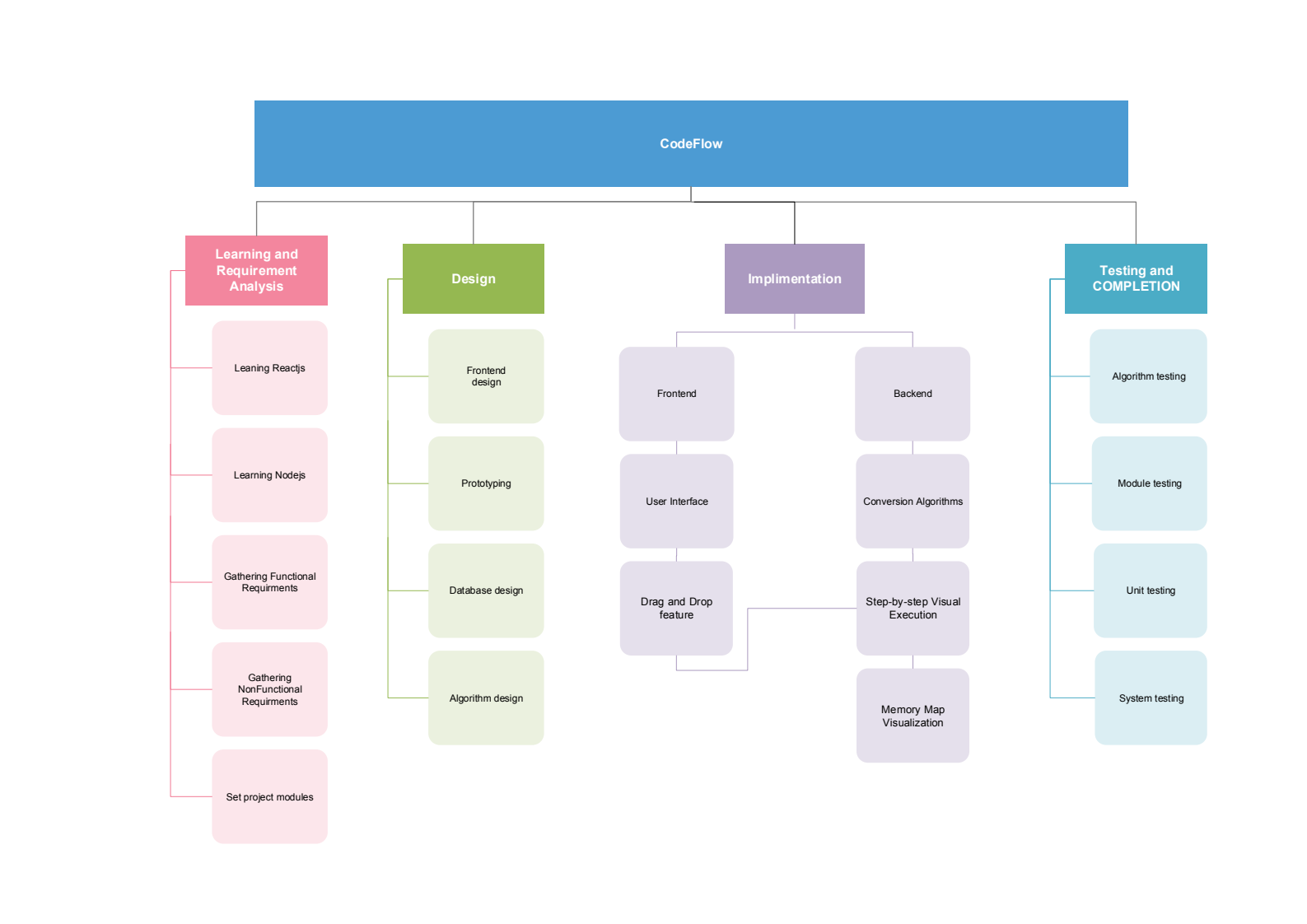
**5. Memory Map Visualization During Execution:**

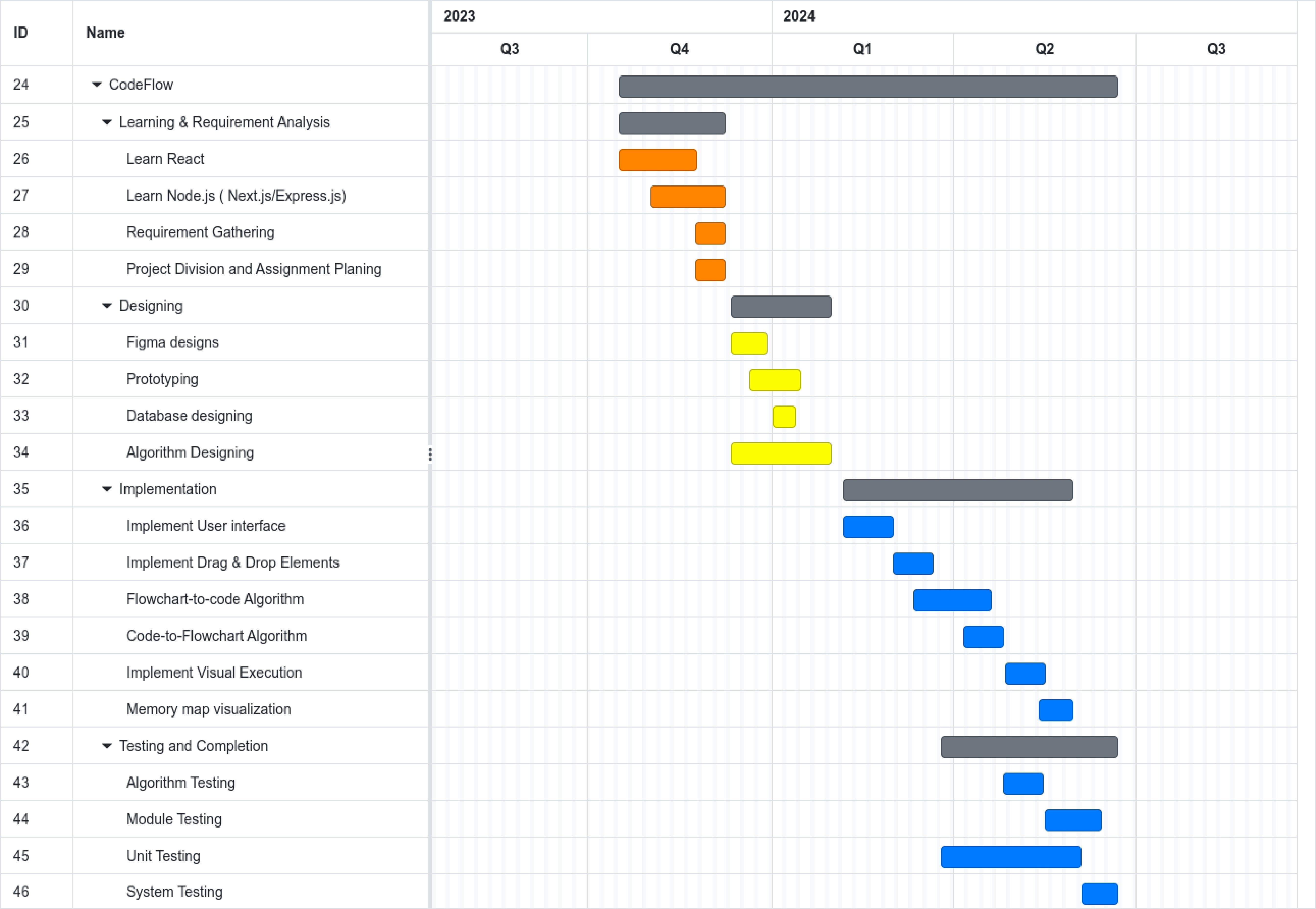
**Frontend and Backend Development:**

* Design and develop features for visualizing memory allocation and manipulation.
* Create graphical representations of memory storage.
* Update memory maps in real-time during program execution.

**Project Plan / Project Schedule / Project Timetable / Project Calendar**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Task** | **Name** | **Start Date** | **End Date** | **Duration** |
| 1 | CodeFlow | Oct 16, 2023 | Jun 21, 2024 | 180 days |
| 2 | Learning & Requirement Analysis | Oct 16, 2023 | Dec 08, 2023 | 40 days |
| 3 | Learn React | Oct 16, 2023 | Nov 24, 2023 | 30 days |
| 4 | Learn Node.js ( Next.js/Express.js) | Nov 01, 2023 | Dec 08, 2023 | 28 days |
| 5 | Requirement Gathering | Nov 23, 2023 | Dec 08, 2023 | 12 days |
| 6 | Project Division and Assignment Planing | Nov 23, 2023 | Dec 08, 2023 | 12 days |
| 7 | Designing | Dec 11, 2023 | Jan 30, 2024 | 37 days |
| 8 | Figma designs | Dec 11, 2023 | Dec 29, 2023 | 15 days |
| 9 | Prototyping | Dec 20, 2023 | Jan 15, 2024 | 19 days |
| 10 | Database designing | Jan 01, 2024 | Jan 12, 2024 | 10 days |
| 11 | Algorithm Designing | Dec 11, 2023 | Jan 30, 2024 | 37 days |
| 12 | Implementation | Feb 05, 2024 | May 30, 2024 | 84 days |
| 13 | Implement User interface | Feb 05, 2024 | Mar 01, 2024 | 20 days |
| 14 | Implement Drag & Drop Elements | Mar 01, 2024 | Mar 21, 2024 | 15 days |
| 15 | Flowchart-to-code Algorithm | Mar 11, 2024 | Apr 19, 2024 | 30 days |
| 16 | Code-to-Flowchart Algorithm | Apr 05, 2024 | Apr 25, 2024 | 15 days |
| 17 | Implement Visual Execution | Apr 26, 2024 | May 16, 2024 | 15 days |
| 18 | Memory map visualization | May 13, 2024 | May 30, 2024 | 14 days |
| 19 | Testing and Completion | Mar 25, 2024 | Jun 21, 2024 | 65 days |
| 20 | Algorithm Testing | Apr 25, 2024 | May 15, 2024 | 15 days |
| 21 | Module Testing | May 16, 2024 | Jun 13, 2024 | 21 days |
| 22 | Unit Testing | Mar 25, 2024 | Jun 03, 2024 | 51 days |
| 23 | System Testing | Jun 03, 2024 | Jun 21, 2024 | 15 days |





**Resources Required**

Noting

**Miscellaneous**

Nothing