CCP PROPOSAL

GROUP MEMBERS:

- **MISHAL KHALID CT-25156**
- **DIYA RANI CT-25165**

COURSE INSTRUCTOR: **SIR ABDULLAH**

COURSE TITLE: **PROGRAMMING FUNDAMENTALS**

COURSE CODE: **CT-175**

DISCIPLINE: **BCIT**

INSTITUTION: **NED UNIVERSITY OF ENGINEERING &TECHNOLOGY**

1.PROJECT TITLE

THE HANGMAN GAME

2.PROJECT DESCRIPTION

The Hangman Game is an interactive, console-based word-guessing application developed using the C programming language. The game dynamically selects a secret word from a given list of words, and the player attempts to identify the word by entering individual letters. For each incorrect guess, a visual representation of the hangman figure is progressively displayed, indicating the remaining chances. Correct guesses reveal the corresponding letters in the word, while incorrect or repeated guesses decrease the available attempts.

This project is an applied demonstration of critical programming concepts, including:

- Conditional logic (if-else, switch) for decision-making
- Iterative constructs (for, while) for repeated operations
- Arrays and strings for data management
- Modular programming through functions to ensure readability and maintainability
- Randomization using rand() and srand(time(NULL)) for dynamic word selection
- Input validation and case-insensitive handling for enhanced user interaction

The Hangman Game combines algorithmic problem-solving with structured software design to create a robust and engaging learning tool for users, while also demonstrating best practices in programming.

3.PROJECT METHODOLOGY

3.1. ANALYSIS AND PLANNING:

- Define the project objectives, scope, and success criteria.
- Develop a comprehensive list of secret words categorized by complexity.
- Design the logical flow of the game, including word selection, guessing mechanism, and win/loss conditions.

- Conceptualize the visual representation of the hangman figure based on remaining chances.

3.2. DESIGN AND DEVELOPMENT:

- Implement the program using C programming language with standard libraries (stdio.h, stdlib.h, string.h, time.h).
- Develop a modular design incorporating functions for:
  - Displaying the hangman figure dynamically based on remaining chances.
  - Checking correctness of user input and updating the word display.
  - Validating user input to ensure reliability and prevent errors
- Implement random word selection and seed generation for unpredictable gameplay.
- Ensure case-insensitive input handling for consistent user experience.
- Incorporate clear prompts and feedback to guide the player throughout the game.

3.3. TESTING AND OPTIMIZATION:

- Conduct rigorous testing with multiple word sets to verify game logic.
- Validate correctness of hangman figure representation at every stage.
- Test edge cases, such as repeated letters, invalid characters, and boundary inputs.
- Optimize code for efficiency, maintainability, and readability.

3.4. IMPLEMENTATION AND DOCUMENTATION:

- Finalize the interactive console-based application with intuitive flow and visual clarity.
- Provide comprehensive documentation explaining the program structure, algorithms, and potential enhancements.
- Deliver a version that can be extended into GUI-based or networked applications in the future.

3.5. TIMELINE:

- **Week 1-2**: Decide on project idea, define objectives and expected outcomes, list features and assign responsibilities.

- Study relevant C concepts, draw flowchart for game logic, write a pseudocode and design game structure and stages.

- **Week 3-4:** Plan and implement header files, write basic structure of main function, implement case-insensitive input handling, constants and variable declarations.
- **Week 5-6:** Add string functions, test loops and conditions, implement Hangman function (drawing figures) and ensure hangman updates correctly for each wrong guess.
- **Week 7-8:** Run final version of the game, fix potential bugs and optimize code.

3.6. EXPECTED OUTCOMES

- **Functional Game:** A fully working Hangman Game in C with accurate word selection, input handling, and visual representation of remaining chances.
- **Enhanced Programming Competence**: Practical experience for the team by using loops, conditionals, arrays, strings, and functions in a real software project.
- **Improved Problem-Solving and Algorithmic Thinking:** Development of logical reasoning and systematic thinking through designing algorithms to handle guessing logic, input validation, and game state updates. Game will also encourage cognitive engagement and strategic thinking in players.
- **Foundational Framework for Research & Development:** The modular program design will allow future enhancements, such as GUI development, multiplayer features, or integration into digital learning platforms.
- **Potential Research or Analytical Insights:** Project will provide insights into user behavior, guessing strategies, and interaction patterns, which can be explored for educational research or software improvement.

---

4. JUSTIFICATION:

The Hangman Game project addresses both technical and educational objectives:

4.1. TECHNICAL JUSTIFICATION:

- Demonstrates practical application of fundamental programming concepts, including loops, conditional statements, string manipulation, arrays, and modular programming.

- Enhances problem-solving and logical reasoning skills through algorithmic design and execution.

- Provides experience in designing interactive software with dynamic input handling and visual output.

4.2. EDUCATIONAL SIGNIFICANCE:

- Encourages engagement through interactive gameplay, improving cognitive and analytical skills.
- Can serve as a tool to enhance language skills, vocabulary acquisition, and spelling proficiency.

---

5. INDUSTRIALIZATION / COMMERCIAL PRODUCT POTENTIAL

Although developed as a console-based application, the Hangman Game has significant potential for commercial and industrial adaptation:

5.1. GRAPHICAL USER INTERFACE (GUI) IMPLEMENTATION:

- Development of GUI-based versions for desktops, tablets, or mobile platforms to improve usability and user engagement.

5.2. EDUCATIONAL INTEGRATION:

- Incorporation into language-learning applications or gamified educational platforms for vocabulary and spelling enhancement.
- Adaptation for classroom or online learning modules, offering interactive teaching tools.

5.3. GAMIFICATION AND DIGITAL EXPANSION:

- Integration of networked multiplayer functionality for competitive or collaborative gameplay.
- Incorporation of levels, scoring systems, and reward mechanisms to enhance motivation and learning outcomes.
- Distribution as part of educational software packages or standalone digital learning products.

5.4. COMMERCIAL PRODUCT VIABILITY:

- The modular and scalable architecture allows transformation into digital applications, mobile apps, or e-learning tools, targeting students, language learners, and casual users alike.
