**Functional Requirement of Morse-Code Convertor:**

* **User Input**:
  + The program should prompt the user to enter a text message to be converted into Morse code.
  + The program should handle both uppercase, lowercase letters and numbers.
* **Morse Code Conversion**:
  + The program should accurately convert each character of the input text into its corresponding Morse code representation.
  + It should handle letters, numbers, punctuation marks, and spaces.
* **Output Display**:
  + The program should display the generated Morse code on the screen or Text.
  + It could use a combination of dots and dashes to represent the Morse code symbols.
* **Morse to Text Conversion**:
  + The program should allow the user to enter Morse code and convert it back into the original text message.
  + It should accurately decode Morse code patterns into their respective characters.
* **Error Handling**:
  + The program should handle invalid input, such as unrecognized characters or malformed Morse code patterns.
  + It should provide appropriate error messages to guide the user.
* **User Interface**:
  + The program should provide a simple and intuitive command-line interface for user interaction.
  + It should display clear instructions for input and output.
* **Efficiency**:
  + The program should be efficient in terms of memory usage and execution speed, even for larger input texts.
* **Code Modularity**:
  + The program's code should be well-organized and modular, with functions for different tasks such as conversion and input/output.
* **Documentation**:
  + The program's source code should be adequately documented, explaining the purpose and usage of functions and important code sections.
* **Testing and Validation**:
  + The program should undergo testing to ensure accurate conversion and proper handling of different input scenarios.
  + It should be validated with various test cases, including edge cases and special characters.
* **Portability**:
  + The program should be written in a portable manner, adhering to the C programming standards to ensure compatibility across different systems and compilers.
* **User-Friendly Messages**:
  + The program should provide informative and user-friendly messages, guiding the user on how to interact with the program.
* **Configurable Settings**:
  + The program could allow users to configure settings such as audio tones, playback speed, and default conversion mode (text to Morse or vice versa).
* **User Input Validation**:
  + The program should validate user input to prevent buffer overflows or unexpected behaviour.