Battleship Project Guide

**Project Description:** Create a Battleship Game using Python that allows users to play the classic naval combat game against a computer opponent. The system will utilize dictionaries to track positions of ships and implement CSV files to store information about the game progress.

# Phase 1: Planning

### Project Objective

* + Clearly define the goal of the project.

|  |
| --- |

### Hardware Requirements

* + Identify the necessary hardware components, including sensors and peripherals.

|  |
| --- |

### Visualization

* + Define the scope of the project, including any additional features.

|  |
| --- |

### Timeline

* + Create a timeline that outlines the estimated duration for each phase of the project.

| **Project** | **Status** | **Notes** |
| --- | --- | --- |
|  | Not started |  |
|  | In progress |  |
|  | Complete |  |
|  | Not started |  |
|  | Not started |  |

### Resource Requirements

* + Identify the tools and libraries needed.

|  |
| --- |

# Phase 2: Design

### User Interface Design

* + Define the structure of the music library system using appropriate data structures. Consider using arrays or linked lists to represent the collection of songs.

|  |
| --- |

### Algorithm Design

* + Define the structure and logic of the application, including how user input will affect the program. (Pseudocode or Flowchart)

|  |
| --- |

### Data Abstraction

* + Develop the procedures/functions.

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |

### Error Handling

* + Plan how you will handle potential errors.

|  |
| --- |

### Test Cases

* + Develop a set of test cases that will be used during the testing phase to ensure the app works as expected.

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |

# 

# Phase 3: Testing and Feedback

### Unit Testing

* + Test individual components of the weather station app, such as sensor connection, sensor data, input validation, and error handling.

|  |
| --- |

### Integration Testing

* + Verify that all components work together cohesively within the app.

|  |
| --- |

### User Testing

* + Invite potential users to test the app and provide feedback on its usability and functionality.

|  |
| --- |

### Bug Fixing

* + Address any issues or bugs identified during testing.

|  |
| --- |

# 

# 

# Phase 4: Documentation

### User Manual

* + Create a user manual that explains how to use the calculator app, including instructions on entering numbers, selecting operations, and interpreting the results.

|  |
| --- |

### Code Documentation

* + Document the code, including comments that explain the purpose of each function and segment of code.

|  |
| --- |

### Project Report

* + Write a comprehensive project report that summarizes the planning, design, and testing phases, as well as the skills and objectives achieved in the project.

|  |
| --- |

# 

# Reflection

* **Python Programming:**
  + How did you utilize Python in implementing the Battleship game?
  + What specific Python features or functions did you use to implement the game logic and decision making?
  + Did you encounter any challenges while programming in Python for this project? If so, how did you overcome them?
* **Data Structures:**
  + How did you represent the game board and ship positions using dictionaries?
  + What advantages did using dictionaries provide in terms of tracking ship positions efficiently?
  + Did you consider using any other data structures for this project? If so, why did you choose dictionaries over them?
* **CSV Files:**
  + How did you utilize CSV files to store and retrieve game progress data?
  + What specific operations did you perform on the CSV files to store and retrieve information?
  + Did you encounter any challenges or limitations while working with CSV files? If so, how did you address them?
* **User Interface (UI):**
  + How did you develop the text-based UI for user interaction in the Battleship game?
  + What features did the UI include to enable users to input coordinates and display the game board?
  + Did you consider implementing a graphical user interface (GUI) instead of a text-based UI? If so, why did you choose a text-based UI?
* **Algorithmic Understanding:**
  + How did you demonstrate your understanding and application of game logic for placing ships, checking hits, and determining the winner?
  + What specific algorithms or decision-making processes did you implement for these game mechanics?
  + Did you encounter any challenges in implementing the game logic? If so, how did you overcome them?
* **Troubleshooting:**
  + How did you implement error handling mechanisms for potential issues during gameplay, such as invalid user inputs?
  + What specific error scenarios did you consider and how did you handle them?
  + Did you encounter any bugs or errors during the testing phase? If so, how did you troubleshoot and resolve them?
* **Project Phases:**
  + How did you plan the implementation of the Battleship game, considering the project goals and data structures?
  + What steps did you take in the design phase to develop the basic structure of the game and the UI?
  + How did you approach testing and gathering feedback for the game? How did you incorporate the feedback into your revisions?
* **Documentation:**
  + How did you provide comprehensive setup instructions for running the Battleship game?
  + What explanations and comments did you include in the Python code to make it more understandable?
  + Did you develop troubleshooting guidelines for common issues that may arise during gameplay? If so, what were they?
* **Python Libraries:**
  + Which standard Python libraries did you utilize for file I/O operations and game logic?
  + How did you use the CSV module for reading and writing CSV files in the context of the Battleship game?
  + Did you consider using any additional libraries for enhanced user interaction? If so, which ones and why?