OBJECT-ORIENTED PROGRAMMING (OOP)

BSCS – 2C

GROUP #3: TIC-TAC-TOE

TEAM MEMBERS AND CONTRIBUTIONS:

- CHARLES CRISTIAN SALTING = LEAD DEVELOPER AND BACKEND DEVELOPER
- CHEZKO JOMREY TUPAS = CO-LEAD DEVELOPER AND FRONTEND DEVELOPER
- MARC ANTHONY BUNAN = PROJECT ASSISTANT AND GAME DESIGNER
- ALORICH DAYRIT = PROJECT ASSISTANT AND JAVASCRIPT DEVELOPER

OOP Implementations

1. Encapsulation:

- Private properties in 'BaseConfig', 'CpuPlayer', and 'Board' keep internal state safe. Accessed via getters/setters (e.g., 'isMusicEnabled()', 'getCells()').

2. Inheritance:

- 'HumanPlayer' and 'CpuPlayer' extend the 'Player' base class; 'GameConfig' extends 'BaseConfig'.

3. Abstraction:

- Abstract classes/methods define contracts:
    - 'Player' declares 'getType()' for polymorphic behavior.
    - 'BaseConfig::validate()' requires concrete validation logic.

4. Polymorphism:

- 'HumanPlayer' and 'CpuPlayer' implement 'getType()' differently. 'Game' accepts any 'Player' subtype, enabling mode flexibility without changing the game's aggregation logic.