Project 2 Analysis and Design

- **Analysis:**
  - Domain (what the project is about):
    - Entities (things in the problem):
      - Boat
      - BoatType (sailing or power)
      - Fleet (a list of boats)
      - Fleet Management System (the main program that runs everything)
    - Actors:
      - User (types menu commands, enters names, expenses, CSV lines)File system (the computer's files we read and write)
  - Function Points:
    - Things done by actors:
      - Print the list of boats
      - Add a new boat
      - Remove a boat
      - Spend money on a boat
      - Exit the program
    - Actions done by the system:
      - Load boats from CSV the first time
      - Load boats from a saved file on later runs
      - Save boats in a file when exiting
      - Store boats in an ArrayList
      - Search for boats by name
      - Check if spending is allowed (can't spend more than the boat cost)
  - Scenarios
    - First run (CSV file)
      - 1.) Program starts with a CSV file name.
      - 2.) Reads each line from the CSV.
      - 3.) Creates Boat objects.
      - 4.) Stores them in an ArrayList.
      - 5.) Shows the menu.
      - 6.) When exiting, saves all boats into a .db file.
    - Later run (no CSV file)
      - 1.) Program starts with no arguments.
      - 2.) Loads the saved .db file.
      - 3.) Shows the menu.
      - 4.) On exit, saves again.
    - Print boats
      - 1.) User chooses
      - 2.) Program shows every boat
      - 3.) Then shows total money paid and total spent

        - Add a boat
            - 1.) User chooses A
            - 2.) Types a CSV-style line
            - 3.) Program splits the line, makes a new Boat, and adds it
        - Remove a boat
            - 1.) User chooses R
            - 2.) Types the name of a boat
            - 3.) Program finds it (case-insensitive)
            - 4.) Removes it or prints “cannot find boat”
        - Spend money on a boat
            - 1.) User chooses E
            - 2.) Types the boat name
            - 3.) Types the amount
            - 4.) Program checks if adding that amount would exceed the purchase price
            - 5.) If okay -> updates expenses
            - 6.) If not -> tells how much money is left
        - Exit
            - 1.) User chooses X
            - 2.) Program saves everything
            - 3.) Program ends
- Design;
    - Classes
        - Boat
        - BoatType (enum)
            - SAILING
            - POWER
        - FleetManagementSystem
            - Scanner
            - File loading/saving
            - Menu
            - ArrayList of boats
    - Data
        - Class Boat (object data - per boat)
            - BoatType type
            - String name
            - Int year
            - String makeModel
            - Int lengthFeet
            - Double purchasePrice
            - Double expenses
    - Methods
        - Boat
            - Constructor (sets up the boat)

        - fromCsv(String line) (creates a Boat from a CSV line)
        - Getters
        - addExpense(double amount)
        - remainingBudget()
        - toString() (returns a formatted line for printing)
- FleetManagementSystem
    - main
    - loadFromCsv
    - loadFromDb
    - saveToDb
    - runMenu
    - printFleet
    - addBoat
    - removeBoat
    - processExpense
    - findBoatByName