Design Details:

We define a Bird class as the base class to hold common attributes for all bird classifications:

birdType: The type of bird.
definingCharacteristic: A defining characteristic of the bird.
extinct: A boolean indicating whether the bird is extinct (true or false).
numWings: The number of wings
The preferredFood attribute is a list that stores the types of food the bird prefers to eat.
Methods (setPreferredFood and addPreferredFood) to set or add food items to this list.

We create specific subclasses for different bird classifications, such as BirdsOfPrey, FlightlessBird, Owls, Parrots, Pigeons, Shorebirds, and Waterfowl. Each subclass inherits from the Bird class and may have additional attributes that are specific to that classification.

Test Plan:

Test individual classes and methods to ensure they work correctly.
Tests:
For each class, we will write unit tests to check the working of constructors, setters, getters, and any other methods.
We can test edge cases, such as creating bird instances with extreme values or invalid input.

Test interactions between different classes.
Tests:
We will create test scenarios that involve multiple bird instances.
We can Test how subclasses interact with the base Bird class and ensure that inheritance and method overriding work as expected.

Test the overall functionality of our program.
Tests:
We will design test cases that cover typical use cases of our bird tracking program.
For example, create bird instances of various types, set their attributes, and perform operations like setting preferred food.

Examples:

Test Case 1: Create Bird Instances

Condition: Testing the creation of instances for different bird classifications.
Example Data:
Create an instance of BirdsOfPrey with "Hawk" as the bird type and other relevant attributes.
Create an instance of Owls with "Barn Owl" as the bird type and other relevant attributes.
Create an instance of Parrots with "African Grey Parrot" as the bird type and other relevant attributes.
Expected Outcome: Confirm that instances of different bird classifications can be created successfully.

Test Case 2: Set and Get Preferred Food

Condition: Testing the setting of preferred food for a bird.
Example Data:
Create a Parrots instance and set its preferred food to a list containing "Seeds," "Fruits," and "Nuts."
Expected Outcome: Verify that the preferred food list is correctly set.

Test Case 3: Test Extinct Birds

Condition: Testing whether the extinct attribute correctly represents whether a bird is extinct.
Example Data:
Create instances of extinct and non-extinct birds, such as a Pigeons instance (extinct) and a Parrots instance (non-extinct).
Expected Outcome: Verify that the extinct attribute reflects the correct status for each bird.

Test Case 4: Test Inheritance and Method Override

Condition: Testing that subclasses inherit attributes and methods from the Bird class and override them as needed.
Example Data:
Create a BirdsofPrey instance with attributes specific to birds of prey and use methods inherited from the Bird class.