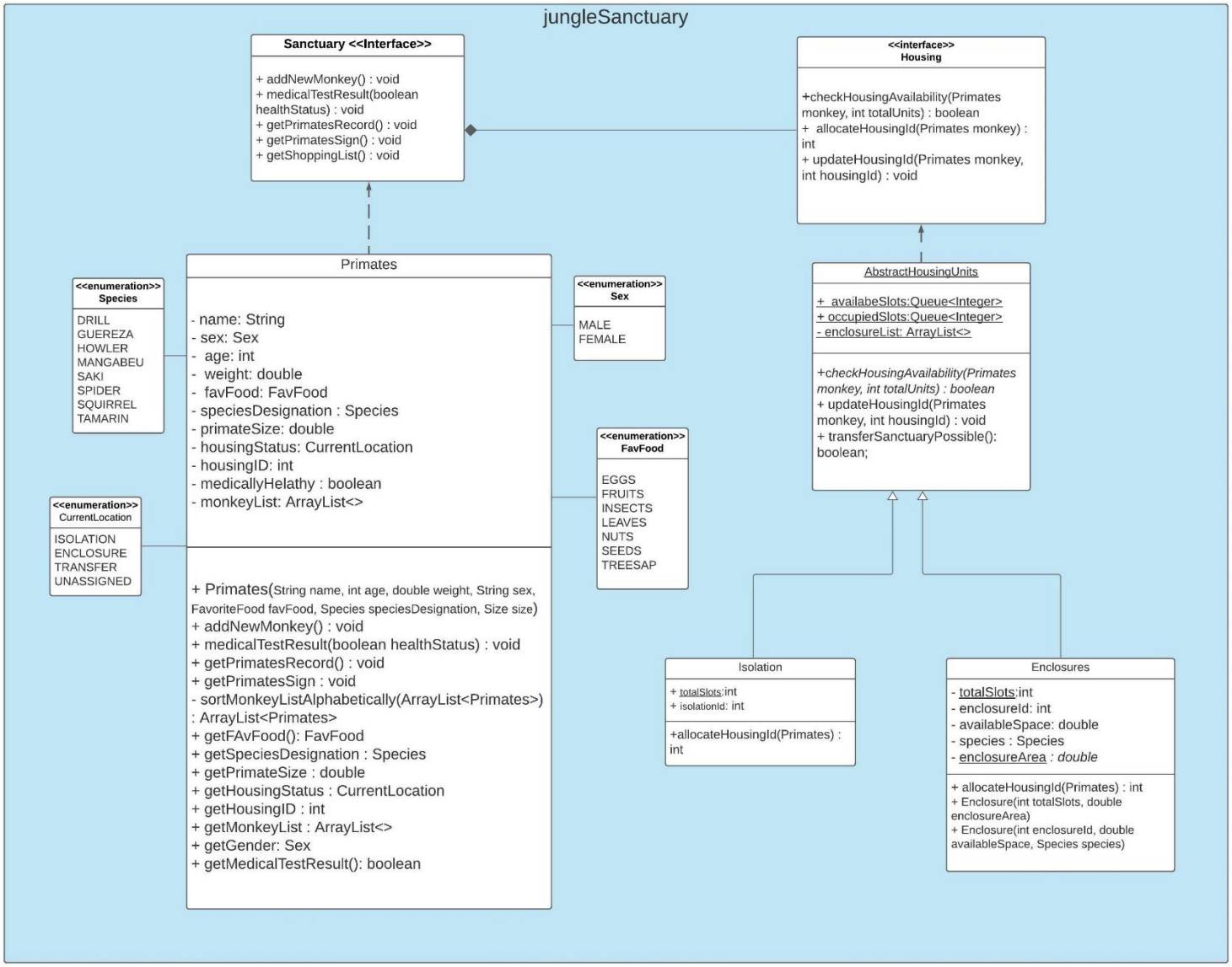
PROJECT 1 - DESIGN MEETING

Primates Sanctuary

UML Diagram:



Notes:

The above picture is the UML diagram for the Primates project.

1. The project design consists of two Interfaces: **Sanctuary** and **Housing**.
2. All the method which client would typically use (related to Primates) are present in the Sanctuary interface and are implemented in the **Primates** class.
3. Enums are being used to store the values that are constant strings like the Species Designation, Favorite Food, Sex and Location Status.

   Species - DRILL GUEREZA HOWLER MANGABEU SAKI SPIDER SQUIRREL TAMARIN
   FavFood - EGGS FRUITS INSECTS LEAVES NUTS SEEDS TREESAP
   Sex – MALE, FEMALE
   CurrentLocation- ISOLATION, ENCLOSURE, NOT ASSIGNED, TRANSFERRED
4. Primates class has private variables representing:

   name
   age
   sex
   speciesDesignation
   weight
   primateSize
   favFood
5. Apart from the variables provided in the problem statement, I also have few more variables like

   housingStatus – this represents a String variable holding the currentLocation of Monkey
   Acceptable Values are: ISOLATION, ENCLOSURE, NOT ASSIGNED, TRANSFERRED
   housingID – this represents Int variable providing us with the unit number of Isolation or Enclosure where the monkey is currently held. Since Isolation and Enclosure are mutually independent values, housingID is same variable for both isolation and enclosure. The combination of housingStatus and housingID would give exact location on the moneky.
   medicallyHealthy – this is a Boolean value to represent if the primate has passed the medical test in isolation center.
   monkeyList – It contains list of all the currently present in Sanctuary.
6. Methods in primate class are below mentioned

	Method Name	Description	Input parameters	Output
1.	Primates(all parameters)	Constructor to create a new primate.	**String name** **int age** **Sex sex** **Species speciesDesignation** **double weight** **double primateSize** **FavFood favFood**	New Primate with all the given parameters get created.
2.	addNewMonkey()	As soon as the primate object is	None	List gets appened with new primary.

		created, it is added to the list of already existing monkeys.		
3	medicalTestResults(boolean)	Method to pass or fail a primate's health checkup	True/False	Updates the **medicallyHealthy** variable of the primate with true or false passed.
4	getPrimatesRecord()	Returns primate along with the location they are held at	None	["Alex, "ISOLATION01", "Paula, Enclosure"]
5	getPrimatesSign	Returns primate's name along with the fav food and sex	None	["Alex, Eggs, Male"]
6	sortMonkeyListAlphabetically	Sorts the list of monkeys alphabetically based on monkey's name	None	None
7	getFavFood	To get primate's fav food.	None	Returns enum fav food of the primate
8	getSpeciesDesignation	To get primate's species type	None	Returns enum species type of the primate
9	getPrimateSize	Returns primate's size	None	Returns size of the primate
10	getHousingStatus	Returns primate's current location	None	Returns which type of housing the primate is held at
11	getHousingID	To get primate's location id of isolation or enclosure	None	Returns unit ID
12	getMonkeyList	To get monkey collection	None	Get all monkeys list.

13	getMedicalTestResult	To get if the primate is passed health check or not	None	Returns if the primate is passed health check or not
14	getShoppingList	Return list of food to be bought along with the quantity		[Eggs : 300, Treesap : 25]…

Methods in AbstractHousingUnits, Isolation and Enclosure classes:

	Method Name	Description	Input parameters	Output
	checkHousingAvailability()	Checks if there is availability in isolation or enclosure as and when required.	Primate object and totalSize	True/false
	updateHousingID	Method to update the housingID field of the primate with unit number	Primate object and unit id	the housingID field of the primate with unit number
	allocateHousingID	Method to assign available housing a number		Returns the integer ID
	Enclosure (int totalSlots, int area())	To initiate the total area and area of enclosure in the begining	Total number of slots and area of each enclosure	Creates a new object with totalSlots(static) and area()

Testing Plan

Primates Class Test cases

Testing Primate Class	Method to be called / Input	Expected Value
Printing the sign	getPrimatesSign()	"Name: Alex , Sex: MALE, housingStatus: ISOLATION"
Monkey Species	getPrimatesRecord()	"Name: Alex, Unit:ISOLATION10"
Monkey Gender	getGender()	MALE
Monkey FavFood	getFavFood()	SEEDS
Monkey Housing Status	getHousingStatus()	ISOLATION
Monkey Housing Id	getHousingId()	4
Monkey Medical Test	getMedicalTestResult()	FALSE
Constructor disallows type not defined in enum	Monkey("drill", Species)	IllegalArgument Exception
Constructor disallows type not defined in enum	Monkey("female", Sex)	IllegalArgument Exception
Constructor disallows type not defined in enum	Monkey("seed", FavFood)	IllegalArgument Exception
Constructor disallows type not defined in enum	Monkey("isolation", HousingStatus)	IllegalArgument Exception
Constructor disallows negative value	Monkey(weight:-20)	IllegalArgument Exception
Constructor disallows negative value	Monkey(age:-20)	IllegalArgument Exception

Isolation Class Test Cases

Testing Isolation Class	Method to be called / Input	Expected Value
Constructor disallows negative value	Isolation(-20,monkey)	IllegalArgument Exception
Incoming primate to be added to isolation	checkHousingAvailabilty	Add to isolation if space is available.
Testing checkHousingAvailability() valid case	totalSlots =2 add first primate move to isolation add second primate move to isolation add third primate call checkHousingAvailability	return value = False;
Testing allocateHousingID	Add first primate move to isolation add second primate move to isolation	houseID should be 2;

Testing Enclosure Class

Testing Enclosure Class	Input	ExepctedValue
Constructor with negative totalSlots (invalid)	Enclosure e1(10,-100)	IllegalArgumentException
Constructor with negative totalArea (invalid)	Enclosure e1(-10, 100)	IllegalArgumentException
Constructor with invalid enum value	Enclosure e1(1,5, "monkey")	Enclosure e1(1,5, "monkey")
Testing allocateHousingID	Add first primate move to enclosure add second primate move to enclosure	houseID should be 2;
Testing checkHousingAvailability() invalid case	totalSlots =10 add first primate call checkAvailability	return value = true;
Testing checkHousingAvailability() valid case	totalSlots =2 add first primate move to enclosure add second primate move to enclosure add third primate call checkHousingAvailability	return value = False;
Constructor with negative roralArea (invalid)	Enclosure e1(1,-5, "monkey")	IllegalArgumentException