Concrete Architecture Structure

We used LibGDX for our game development framework, this Java-based software didn't force us to use a specific design and allowed us to create our game how we wanted to while supporting us with many features[1]. As well as cross-platform support which we will not be utilising at least this early in the project it provides Low-Level OpenGL helpers such as Vertex Arrays, Meshes and Tile Map Support all of which we have been utilising early in the development of our project.

To accurately model the architecture of our program we used intelliJ's built in class diagram creator to model for us all the class, methods and attributes of our game. This saved us a lot of time as we didn't have to manually enter data and it gives a perfect representation of all of our written code.

Architecture justification

Classes that have been implemented in assessment 2 version of our game:

Game

Link to requirements: 1.1, 1.3, 1.8
This is where most of the running of the game happens, it handles updating the map after each interaction and resetting the ships data at the end of each turn, tracing if it is player's or enemy's turn. It initially creates a game drawing the map, the ships and everything else that is needed. It handles the inputs of the user to move their ships.
It is a singleton object because there will only ever be one instance of the game object at one time.

Map

Link to requirements: 1.1
The map class is the same as the map class described in Assessment 1, it stores the grid which the map is made up of and is used to update the ships locations using their x and y coordinates. It also holds data for where any departments, colleges or any buildings are on the map. Furthermore it also calculates whether any ship movement is valid, trying to move onto land is invalid. During the implementation, we added more features in the Map class base on our first version of abstract architecture, to make sure the map is more suitable for our game. We added some inner function based on the map to "find the possible move for the ship" in the class so that could be called when needed.

Ship

Link to requirements: 1.3
The ship class contains all the information about the moves that ship has done and where it is as well as if it can actually attack, each ship has its own independent amount of moves and attacks instead of the player having a set number.
There is then the ship list which keeps track of each of the individual objects of the ship class.

Department and College
Link to requirements: 1.1
Two different classes that contain information about the colleges and departments located on the map like their name, health, where they are and how many ships they own. We remove the effect property for college since it is not useful for this deliverable and added properties for the college ships.

Coords
Link to requirements: 1.1, 1.3
Contains all information relating to the coordinate system used in the game with the ability to edit coordinates and retrieve all as a list.

PointSystem
Link to requirements: 1.5
A small but important class that retrieves the points and currency of the player, sets the points or currency and adds points or currency to it. The points are given to a player after their game. The currency will be spent on upgrade or buying objects in game by player (The currency is not implemented yet). This is a requirement for Assessment 2.

ShipList
Link to requirements: 1.3
Another small class that has the ability to generate a new ship randomly and then stores the ships on the map in a list.

EnemyShipAI
Link to requirements: 1.2, 1.3
A class that gives the computer an AI so then they can also enter combat with the players ships. This works by moving towards a ship if it is within its range and can attack it when it is close enough, and move randomly base on its own speed if there is no ship in different team around it. This also give the player an objective of trying to destroy all of the enemy AI ships.

Classes that haven't yet been implemented:
Minimap
Due to not implementing a GUI yet as it was not necessary there is nowhere to put a minimap and so is not possible at this stage, as we create the maps used in the game we can simply limit the size of them to the point where everything is visible on the screen. Later in development the sizes can be drastically increased and the minimap would be needed to know where your ships are in the map; it could also be especially useful when a fog of war is added to the game.

HUD
All of the graphical output is just alterations of the map for the moment, in future iterations of the game there would be a HUD in order to do extra things such as accessing the shop but all of the main functionality can be done without the need for a HUD so we decided to exclude it for assessment 2

Events/minigame

We didn’t include random events for Assessment 2 but did create the groundwork for them to be implemented later with the islands scattered around the map. For them to be fully implemented how we want it ships would need characteristics and health statistics and implementing all this would be too time consuming at this point in development. The minigame is something that is a requirement later on in development as so that would be a better time to implement it rather than now.

Shop/Item

Due to not being a requirement according to the Assessment 2 brief it was excluded in this build of the game so we could focus on the other parts of the game.

Camera

Camera does not have a class of its own but a camera is used to view the map top down.

Appendix

https://libgdx.badlogicgames.com/features.html [1]