Change	Purpose / Justification
ShipUpgrade	Class removed and put as codified variable in DepartmentNode. Complexity of upgrades was considered trivial and it having its own class non essential
CollegeStatus in CollegeNode	Made that related to whether the college has been defeated in battle instead of related to completed quests (we removed Quests)
Game to GameLogic	Functionality was split between GameLogic and GameVisual due to inclusion of graphics. Now GameLogic only contains game data and functions manipulating said game data
Battle to BattleMode and BattleModeVisuals	Split into two classes where BattleMode is the logic and data pertaining to battle and BattleModeVisuals is functional graphical end
Added	**Purpose / Justification**
IsWon to GameLogic	For use by game end visual section of code to identify whether game was won or not
NeighbourNodes to GameLogic	A list of integers which is a local variable of a variable in node for convenience
GetoCollegeObjectives() to GameLogic	Looks at players college and sets players objectives relevant to that college
Obj2Str() to GameLogic	Visualises an objective as a human readable string
NodeMapGenerator() to Node class	Produces list of nodes which forms node map, used within GameLogic. This includes initialising colleges and departments
card of type Card in CollegeNode	Instead of linking to an ID, links directly to card college is selling
bossShip in CollegeNode	Added for ease to be the specific ship fought in battle with a college
bossShip in CollegeNode	Added for ease to be the specific ship fought in battle with a department
upgradeCost in DepartmentNode	Links to removal of ship upgrade
attackCollege() to getBossShip() in Node	To be a getter relevant to the function of the method
buyCard() to getCard() in Node	Instead of linking to card database with card id it links to card associated with a node
attackDepartment() to getBossShip() in Node	To be a getter relevant to the function of the method
buyUpgrade() in	Instead of taking a choice, it is always the same dependent on

DepartmentNode	department
upgrade2Str() in DepartmentNode	Converts the upgrade code to human readable string
effectString2Text() in Encounter	Converts the codified effects to a readable option which is then stored in options
pointsWorth in Ship	Info relating to rewards a player would get for defeating a ship
goldAmount in Ship	Info relating to rewards a player would get for defeating a ship
isSpecial to Ship	So graphically we could know whether ship was independent or a member of a college or department
Ship() to Ship	Constructor that takes no input (not incl in diagram but noted for relevancy)
Ship(totalHealth, currentHealth, totalMana, manaRegion, pointsWorth, goldAmunt, deck, isSpecial) to Ship	This constructor for enemy (non-player) ships
playerMaxMana to Player	Temp variables for managing mana within battle
enemyMaxMana to Enemy	Temp variables for managing mana within battle
updateHealth() to BattleMode	Needed a method to manage health
showManaBar() to BattleMode	Outputs a string relating currentMana compared to totalMana
basicEnemyAI() to BattleMode	Manages enemy behaviour
gameIsOver() to BattleMode	Check to see if the battle is over - doesn't contain actual end game procedure
updateDefence() to BattleMode	Updates the defence modifier used by a target ship by some amount
isCardEmpty() to BattleMode	Checks to see if some slot in the hand is empty
BattleModeVisuals	New class
CollegeVisual	New class
DepartmentVisual	New class
GameEndVisual	New class
GameVisual	New class
ShipVisual	New class
Every visual class has Game game (LibGDX game)	New class

Every visual class (except GameVisual and GameEndVisual) has parent	Screen adapter GameVisual does not have parent as it is the root screen - parent to all others GameEndVisual does not have parents because its a dead end screen
Every constructor will contain game and if it has parent contains parent	Game and parent are both used for screen shifting, allows us to go from each visual screen to another.
Most screens also contain GameLogic	GameLogic is initialised in the root screen and used throughout all other screens as it manages the data used by all other screens throughout the game
Every visual class will contain some stage	Because stages are what allow us to put visual elements on the screen
GameVisual has set of resource labels	displaying data stored in game logic
shipButton to GameVisual	has which is textbutton Leads to ShipVisual which displays objectives and ship stats
nodeButtons	List of textbuttons which displays the node map, neighbouring nodes, current node and allows the player to choose where to traverse or open up the college or department options should the player be on one of these
Set of textbutton.textbutton styles	Used to differentiate between current neighbouring and non-neighbouring nodes within the node map
Every visual screen adapter contains a overridden render method	This draws the screen elements onto the screen after clearing it, render method will also include anything that needs to be evaluated on screen change, such as if the player has lost game due to an encounter
addBackground() to GameVisual	Add background image
createNodeMap() to GameVisual	Creates all the labels in nodeButtons, gives them the relevant textbutton styles and puts them on the screen according to coordinates. This also includes adding different click listeners, dependent on whether it is a collegeNode, departmentNode or neither
createTopLabel() and updateTopLabel() to GameVisual	Initialises and updates the text of the labels we use to visualise resources
turnChange() to GameVisual	Evaluates what goes on during a turn change within GameLogic, it traverses the node in GameLogic then checks the type of the currentNode and goes to different screens dependent of the type. It also checks if traversing the node causes you to win or lose the game

updateNodeMap(0 ot GameVisual	Changes the label styles of specific nodes depending on its current, neighbour or neither node. This is needed for currentNode changes when a node is traversed
endGame() in GameVisual	Screen changes to GameEndVisual
encounter of type Encounter in EncounterVisual	Contains an encounter called Enouncounter which contains encounter that its showing
optionButtons in EncounterVisual	Allows you to pick an option from Encounter
descriptionLabel in EncounterVisual	Displays description
createOptionButtons() in EncounterVisual	Iterates through all the options and creates the relevant one
interpretEffect() in EncounterVisual	Takes in chosen effect, interprets it into the system, executing the choice that has been made (eg. chose to lose five gold, decreased currentGold by 5)
collegeNode in CollegeVisual	Contains college you're working in,
cardButton in CollegeVisual	Button which allows player to buy a card
departmentNode in DepartmentVisual	
upgradeButton in DepartmentVisual	Button which allows player to upgrade the ship
constantUpgradeCost in DepartmentVisual	Constant is included to make it accessible within the click listener
nameLabel in CollegeVisual and DepartmentVisual	Displays name and allied status of the relevant college/department
supplyButton in CollegeVisual and DepartmentVisual	Allows user to buy supplies
clickThis in CollegeVisual and DepartmentVisual	Which holds the parent screen adaptor and with clickThis it allows the battles to switch screens to the main screen as opposed to the college/department screen
attackButton in CollegeVisual and DepartmentVisual	User fights a specific bossShip for relevant college/department. When it creates the battle it uses
exitButton in CollegeVisual and DepartmentVisual	Allows player to screen switch from college/department without doing anything
shipUpgradeInterpreter() in DepartmentVisual	Reads the upgrade of current department and executes appropriately

battleMode in BattleModeVisual	Controller for the battle
handTextures in BattleModeVisual	List of textures, contains 4 textures of the cards in the hand
backgroundTexture in BattleModeVisual	Texture of the backgrounds
shipTexture in BattleModeVisual	List of textures contains textures of the ship as it degrades
exitButton in BattleModeVisual	Spawns when battle is ended and allows screen to switch
gold/score labels in BattleModeVisual	Displays gold/score gained
updateHand() in BattleModeVisual	Updates the hand when the cards in the hand change
chooseCard() in BattleModeVisual	Chooses card and changes textures
endBattle() in BattleModeVisual	Spawns exitButton and gold/score and officially ends the battle in battle mode
updateShip() in BattleModeVisual	Updates the ships texture in accordance to its health
isWon in GameEndVisual	Checks if the game is won
Score in GameEndVisual	Displays score
Ship of type Ship in ShipVisual	Contains a set of labels relating to the ships stats
objectiveLabels in ShipVisual	List of labels displaying objectives and whether they have been completed or not
exitButton in ShipVisual	Allows player to leave the screen
AllHandsOnDeckGame	Contains itself and instance of GameVisuals that it instantiates when it changes to that screen
render() in AllHandsOnDeckGame	contains super.render() to ensure everything gets rendered
Removed	**Purpose / Justification**
Menu	It was considered a non-essential feature for this portion of development
Quests	Non essential feature
ResourceShop	Considered easier to have functionality of buying supplies kept within college and department nodes as opposed tio linked to separate class

MiniGame	Non essential feature for this stage of development
weather from Node	Considered non essential
visibility from Node(boolean)	Considered non essential
CheckQuestCompletion()	Links to removal of Quests
startBattle() from Encounter	Functionality is now codified within variable effects
enemyShip from Encounter	Ships and battles are codified within effects