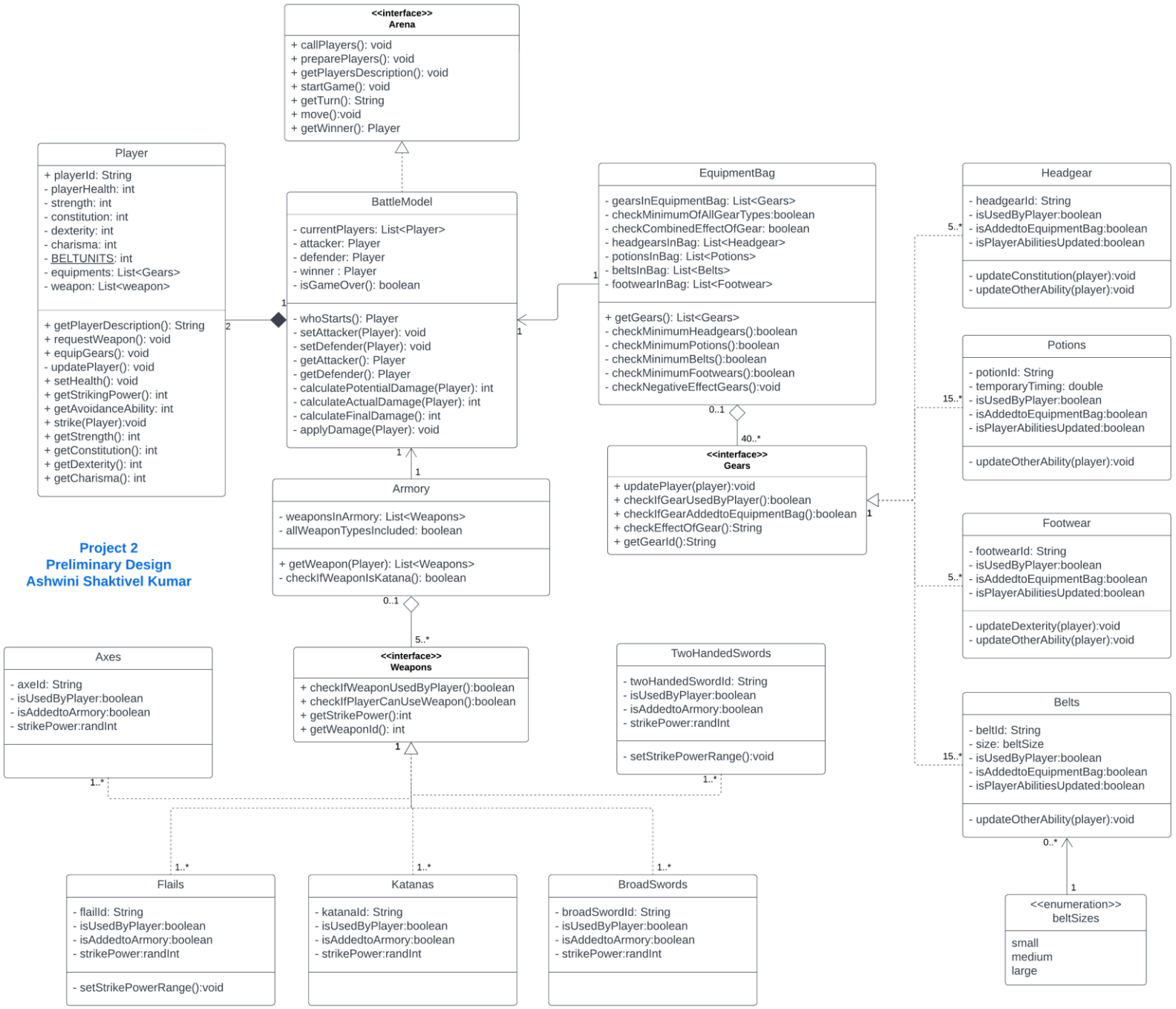
Project 2 – Battle

Preliminary Design - Ashwini Shaktivel Kumar



Testing Plan:

Player:

Testing construction for getPlayerDescription()	getPlayerDescription()	Expected
New player	getPlayerDescription()	Player name: Player 1 Strength = 18 Constitution = 12 Dexterity = 15 Charisma = 9 Gears = [] Weapon = []
Player equipped with gears and weapon	getPlayerDescription() after requestWeapon() and equipGears()	Player name: Player 1 Strength = 13 Constitution = 18 Dexterity = 11 Charisma = 12 Gears = [Headgear1, Potion1, Potion3, Potion5, Potion6, Belt3, Belt5, Belt6, Footwear4] Weapon = [Katana2, Katana3]

Testing construction for getStrength()	getStrength ()	Expected
New player	getStrength ()	18
Player equipped with gears and weapon	getStrength ()	13

Testing construction for getConstitution()	getConstitution ()	Expected
New player	getConstitution ()	12
Player equipped with gears and weapon	getConstitution ()	18

Testing construction for getDexterity()	getDexterity ()	Expected
New player	getDexterity ()	15
Player equipped with gears and weapon	getStrength ()	11

Testing construction for getCharisma()	getCharisma ()	Expected
New player	getCharisma ()	9
Player equipped with gears and weapon	getCharisma ()	12

Testing construction for setHealth()	setHealth(int)	Expected
Update health upon attack	setHealth(-8)	Void Health set from 44 to 36

Testing construction for getStrikingPower()	getStrikingPower()	Expected
Check for the attacker's striking power calculated based on strength, gear effect and random number	getStrikingPower()	9

Testing construction for getAvoidanceAbility()	getAvoidanceAbility ()	Expected
Check for the defender's avoidance ability calculated based on dexterity, gear effect and random number	getAvoidanceAbility ()	9

Testing construction for strike(Player)	strike(Player)	Expected
Player 1 is attacker, player 2 is defender.	Player1.strike(Player2)	Player 2 health updated according to damage
Player 2 is attacker, player 1 is defender.	Player2.strike(Player1)	Player 1 health updated according to damage

BattleModel:

Testing construction for callPlayers()	callPlayers()	Expected
Call new players on the arena	callPlayers ()	New player objects created within the battle arena
Testing construction for preparePlayers ()	preparePlayers()	Expected
Prepare players by equipping gears and weapon on the arena	preparePlayers ()	Player description updated

Testing construction for getPlayersDescription()	getPlayersDescription()	Expected
New players	getPlayersDescription()	Player name: Player 1 Strength = 18 Constitution = 12 Dexterity = 15 Charisma = 9 Gears = [] Weapon = [] Player name: Player 2 Strength = 11 Constitution = 8 Dexterity = 20 Charisma = 24 Gears = [] Weapon = []
Players equipped with gears and weapon	getPlayersDescription()	Player name: Player 1 Strength = 13 Constitution = 18 Dexterity = 11 Charisma = 12 Gears = [Headgear1, Potion1, Potion3, Potion5, Potion6, Belt3, Belt5, Belt6, Footwear4] Weapon = [Katana2, Katana3] Player name: Player 2 Strength = 19 Constitution = 9 Dexterity = 19 Charisma = 17 Gears = [Headgear3, Potion2, Potion4, Potion7, Potion10, Belt2, Belt7, Belt9, Footwear2] Weapon = [Axe1]

Testing construction for getTurn()	getTurn()	Expected
Player 1 is attacker, player 2 is defender.	getTurn()	Attacker: Player1 Defender: Player2
Player 2 is attacker, player 1 is defender.	getTurn()	Attacker: Player2 Defender: Player1

Testing construction for startGame()	startGame ()	Expected
Start game when Player 1's charisma is greater than Player 2.	startGame () getTurn()	Attacker: Player1 Defender: Player2
Start game when Player 2's charisma is greater than Player 1.	startGame () getTurn()	Attacker: Player2 Defender: Player1
No players on arena	startGame()	InvalidStateException

Testing construction for move()	move()	Expected
Player 1 is attacker, player 2 is defender.	move()	Player 2 health updated according to damage
Player 2 is attacker, player 1 is defender.	move()	Player 1 health updated according to damage

Testing construction for getWinner()	getWinner ()	Expected
Player 1 has health less than or equal to 0.	getWinner ()	Player2
Player 2 has health less than or equal to 0.	getWinner ()	Player1

Armory:

Testing construction for getWeapon(Player)	getWeapon (Player)	Expected
Weapon assigned to the player	getWeapon (Player1)	Weapon = [Katana2, Katana3]

EquipmentBag:

Testing construction for getGears(Player)	getGears (Player)	Expected
Gears assigned to the player	getGears (Player1)	Gears = [Headgear1, Potion1, Potion3, Potion5, Potion6, Belt3, Belt5, Belt6, Footwear4]