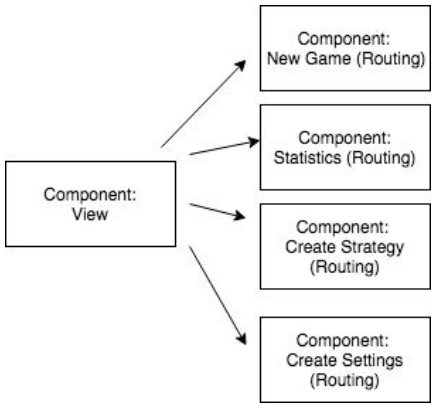
Incremental Testing and Regression Testing

Team 20 | Thomas Chen, Kelvin Choi, Scott Merritt, Aaron Althoff, Dan Morton

Classification of Components [Mostly the same as Sprint 1]

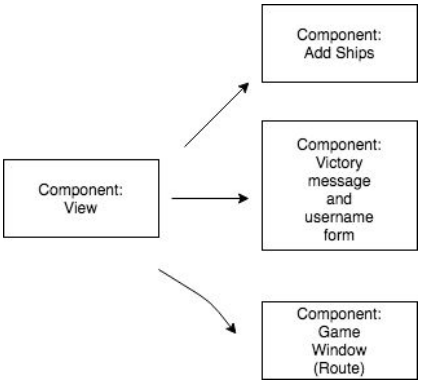
Menu Module:

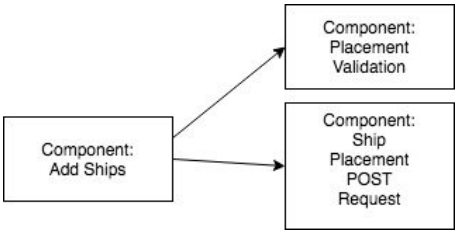
Inputs: None
Outputs: None
Dependent Components: New Game, Statistics, Create Strategy, Settings (Target Destinations for buttons)



New Game Menu Module

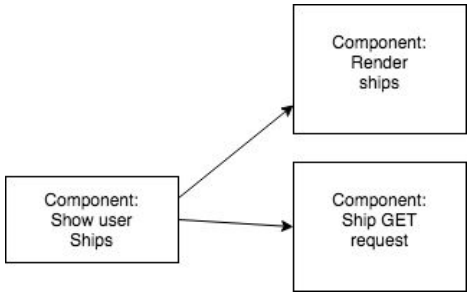
Inputs: Username (string), Victory Message (string), Ship Placement
Output: Username, Victory Message, Ship Placement
Dependent Components: Victory message and username form, Game Window (Target Destination), Add ships, Placement Validation, Ship Placement POST request

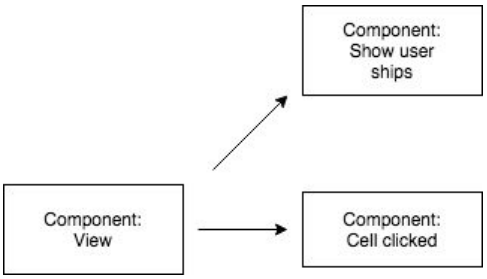


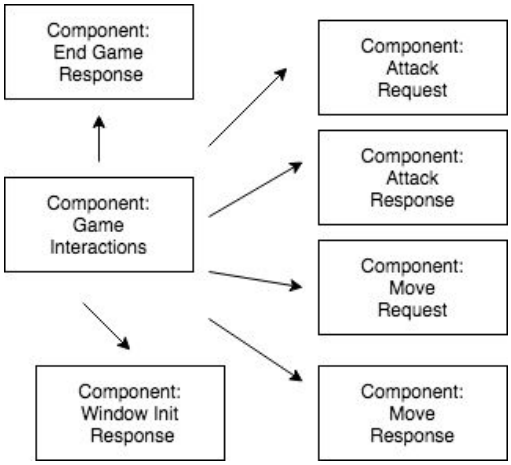


Game Window Module

Inputs: Player ship locations (ship array of where ship is an array of 2d coordinates)
Player victory message (string)
Player username (string)
Dependent Components: Show user ships, Cell Clicked, Render Ship, Ship GET request, Attack Request, Attack Response, Move Request, Move Response

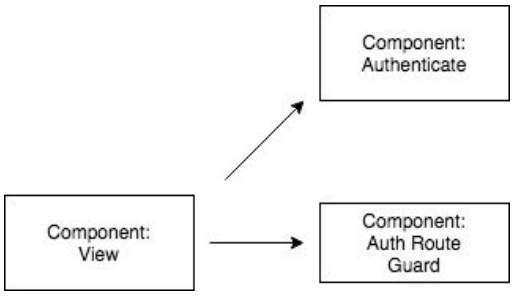






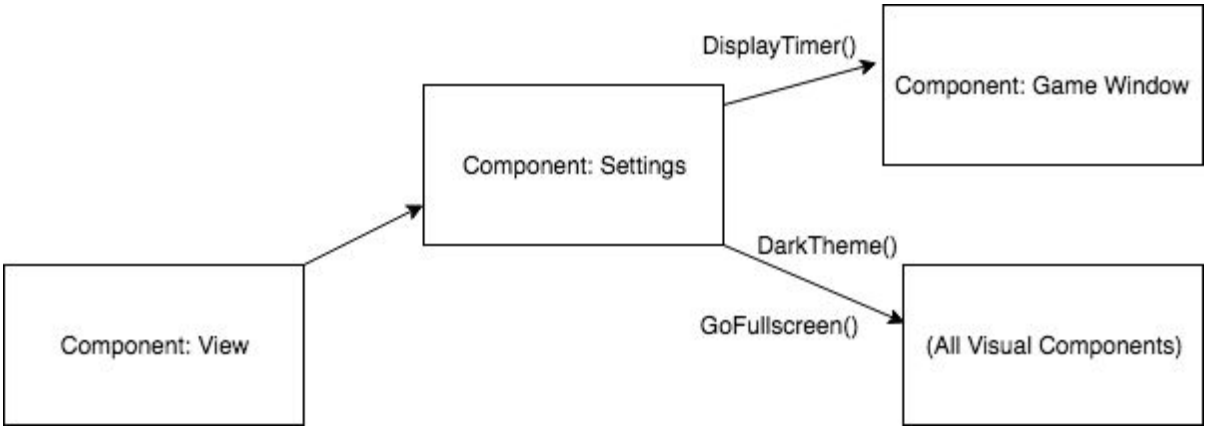
Authentication Component

Inputs: Google sign in credentials
Outputs: Authentication guard unlocked
Dependent Components: Authenticate, Auth Route Guard



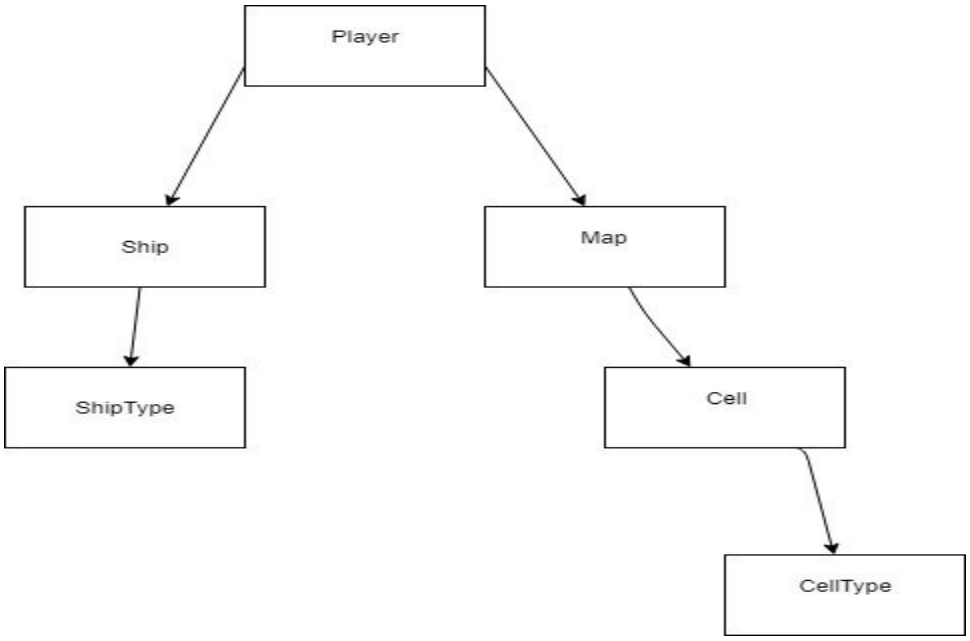
Settings Component

Outputs: DarkTheme [True/False], Fullscreen [T/F], Timer [T/F]
Dependent Components: Game Window Module (with/without Timer), all components' displays are affected by DarkTheme and Fullscreen



Model Module

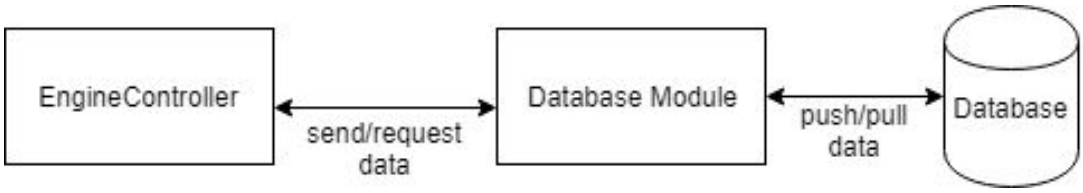
Dependent Components: Ship, Ship Type, Map, Cell, Cell Type



Statistics Module

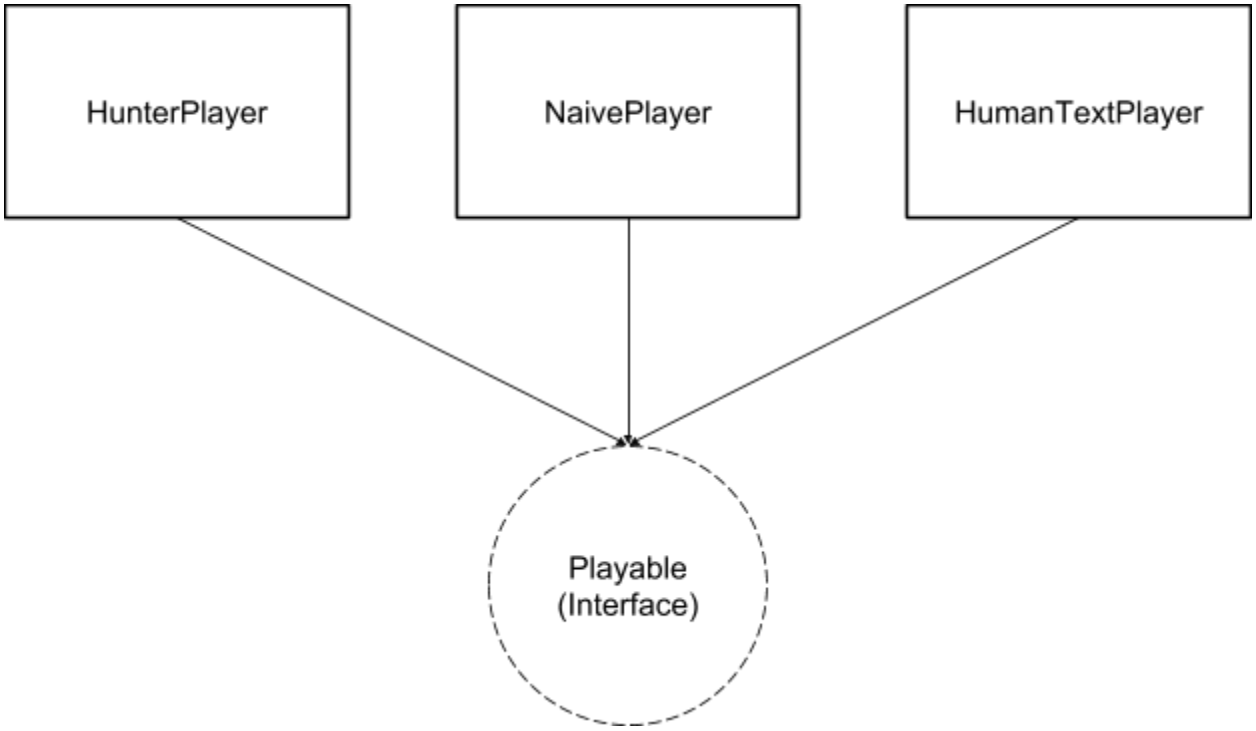
Dependent Components: PlayerStat, GameStat, Statistics

Database Module



AI Module

Dependent Components: Playable, HunterPlayer, NaivePlayer, HumanTextPlayer



Incremental Testing Form [Same as Sprint 1]

For Frontend testing we decided to work with a top-down approach for our incremental tests. We chose to go with this form because we believed it would best suit our working method since we created a MainMenu component first, then gradually began constructing components that would stem from there. As such, we've been able to begin testing with the MainMenu and it's direct components, then progress through our tree of components in order to test the connections between each of the modules.

For Backend testing we actually decided to go with a bottom-up approach, since the backend structure was built starting with the rudimentary modules of the game itself (i.e. Ship, Cell, Grid,

etc.), rather than starting with the MainMenu component and working down. This worked with our system because much of the backend work is dealing with the user playing the game, and as such required the smaller modules to be built first before working up towards building the complete game. We wanted to make sure our testing form followed this, and for that reason we chose a bottom-up approach in our incremental testing.

Module: GameWindow Component

Incremental Testing

Defect No.	Description	Severity	How to corrected
1	AttackResponse would not mark ships that were actually hit. Enemy would perform attack and not get a correct response.	1	The user ships were not being added correctly to the user. Fixed in EngineController.java
2	The game does not end when it is suppose to.	2	Added endgame condition that is triggered by a turn response. This locks the game and disallows any further user interaction until a new game is created.
3	The game does not display the result of each turn	2	Message variable on GameWindow was updated to reflect the value returned from AttackResponse
4	The game was not informing the user when a ship was sunk	2	Added condition in EngineController that informs frontend that a ship was sunk and is reflected in the game windows message.
5	The game was not changing the color of sunk ships to dark red	3	When the frontend receive that a ship was sunk, logic was added to change all the cells associated to that ships background color to dark red.
6	Move request made by user	1	The old ship (its previous

	reflected on front end, however not updating game data correctly on the backend.		location) was not being properly removed to be re-added with a new ship (in the moved ships new location) of the same type. Function removeShip() was adjusted to fix this problem.
7	Game was showing users ships being hit (with black cells) in locations where ships were not present.	1	Tied to defect 6 not properly updating the users grid when a ship move is done.
8	On creating a new game after just playing one game, the users previously placed ships appear on the user grid	1	Reset functionality was added in
9	Game window not performing an enemy move when a user just performed an invalid move ship move	1	Added logic to MoveResponse() that represents the invalidMove response. EngineController.java handles the invalidCase and makes sure to make an enemy move after an invalid user move regardless.

Regression Testing

Defect No.	Description	Severity	How to corrected
1	Game window was not being pushed to from new game menu	1	This defect was due to an error with how the backend was handling responses. To fix this
2	Game window was allowing user to make an attack when they clicked on their own grid	1	In onCellClicked() in game-window.ts, a guard was added to make sure that the origin of the click was not from the user grid.
3	ShipTypes were not properly being mapped to ShipId	1	A remapping was done and was reflected throughout the application.

Module: Database Component

Incremental Testing

Defect No.	Description	Severity	How to corrected
1	SQL Exception was being thrown every time function attempted to add data to the DB	1	SQL query in java class had a SQL syntax error that prevented anything from being added
2	Data being sent from EngineController was not appearing in DB	2	Forgot to add target annotation for function to send data

Regression Testing

Defect No.	Description	Severity	How to corrected
1	New users logging in would not appear in the database	3	URL to send request to backend was incorrect, and UserID given to the functions for the DB was not properly formatted
2	GameData after a game was not being saved into the DB	2	Reconfigured PlayerStats and GameStats classes to work within Game class that was interfaced with the client
3	GameData was not being retrieved correctly for a user in the Statistics page	1	Created a special DBGameStat class to wrap together data from the DB so it could be easily sent to the front end
4	GameData only appears after second call to DB, not first.	3	Fixed issue with asynchronous calls in the backend that were causing a delay in the data being transferred

Module: AI Component

Incremental Testing

Defect No.	Description	Severity	How to corrected
1	HunterPlayer fails execute any moves in test epoch and loses every game.	2	HunterPlayer was using deprecated move function instead of the new turn function, switch to turn function for each turns action.

Regression Testing

Defect No.	Description	Severity	How to corrected
2	HunterPlayer AI not storing state of opponent board and thus unable to carry out any moves.	1	Make sure to call addOpponent() before utilizing HunterPlayer AI otherwise no opponent state can be stored. In the future should embed addOpponent into constructor of AIs.
3	NaivePlayer not making any moves in test epoch and loses every game.	2	Same bug as HunterPlayer in defect 1, requires same fix by switching to new turn() pattern.
4	HumanTextPlayer halting game on first turn.	2	Did not override move method with some defined behavior even though move is called in the game. Fix is to simply override the function and in this case have it naively call attack as well.

Module: Statistics Component

Incremental Testing

Defect No.	Description	Severity	How to corrected
1	Win/Loss ratio being calculated as undefined or negative	1	Was increment win/loss on the inner loop of the traversal of all constituent PlayerStats in GameStat objects should increment in outer loop to fix.
2	Avg hit and miss percentages not adding up to 1.0	1	Was only dividing by number of games but not dividing by number of players as well, resolved by adding the additional division calculation.
3	Match time being calculated incorrectly (underestimate)	2	The match timer was starting after all of the initialization of game components. Moved timer to begin on initialization of game.

Module: Authentication Component

Incremental Testing

Defect No.	Description	Severity	How to corrected
1	After logging out, the client was not allowing for another user to sign in on the login page.	1	Used the correct route direction to go back to login page and user has to refresh page in order to display the “google sign in” button.

Regression Testing

Defect No.	Description	Severity	How to corrected
1	When clicking the logout button, the current page was refreshing every few seconds.	2	Onclick function for logout had an unnecessary function that was removed.
2	User was still logged in after clicking logout button.	1	Get the proper instance of the current user needed to call the log out function.

Module: Settings Component

Incremental Testing

Defect No.	Description	Severity	How to corrected
1	Dark Theme not applied correctly to Game Window	1	Added wrapping container for all Game Window HTML elements and added a dark theme CSS class for the container
2	Timer displaying in New Game component and not Game Window	1	Moved code to display timer from New Game to Game Window

Regression Testing

Defect No.	Description	Severity	How to corrected
1	Timer starts counting from when New Game is opened, not when Game Window is initiated despite	1	Remove code to start counting on the timer from New Game component

	the timer being moved to Game Window. This caused the initial time of the timer to be offset by the amount of time spent in New Game		