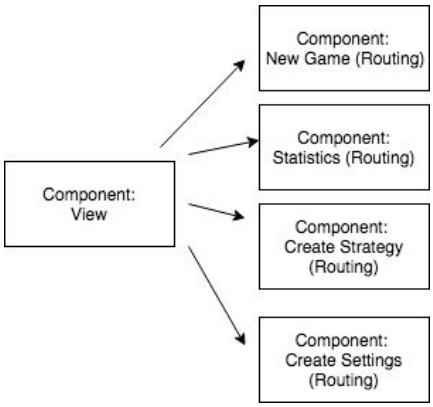
Incremental Testing and Regression Testing

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

Classification of Components

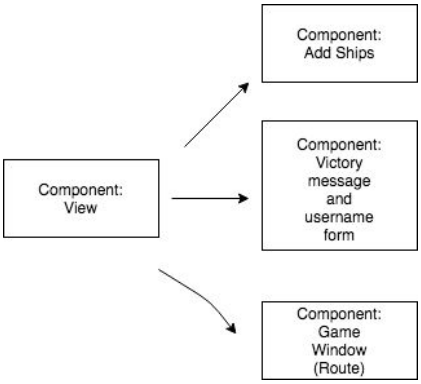
Menu Module:

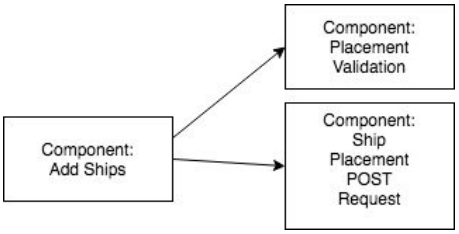
Inputs: None
Outputs: None
Dependent Components: New Game, Statistics, Create Strategy, Settings (Target Destinations for buttons)



New Game Menu Module

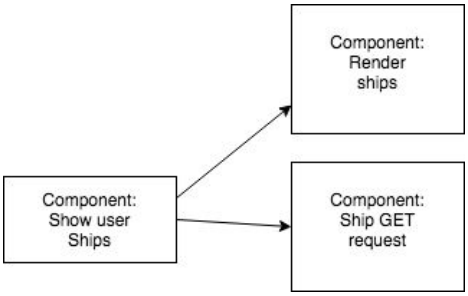
Inputs: Username (string), Victory Message (string), Ship Placement
Output: Username, Victory Message, Ship Placement
Dependent Components: Victory message and username form, Game Window (Target Destination), Add ships, Placement Validation, Ship Placement POST request

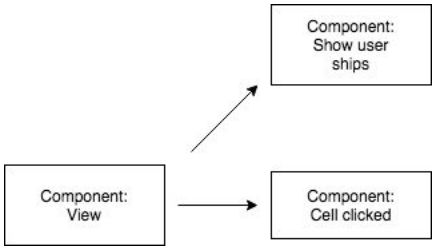




Game Window Module

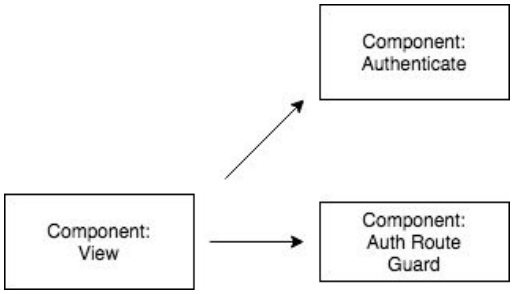
Inputs: Player ship locations (ship array of where ship is an array of 2d coordinates)
Player victory message (string)
Player username (string)
Dependent Components: Show user ships, Cell Clicked, Render Ship, Ship GET request





Authentication Component

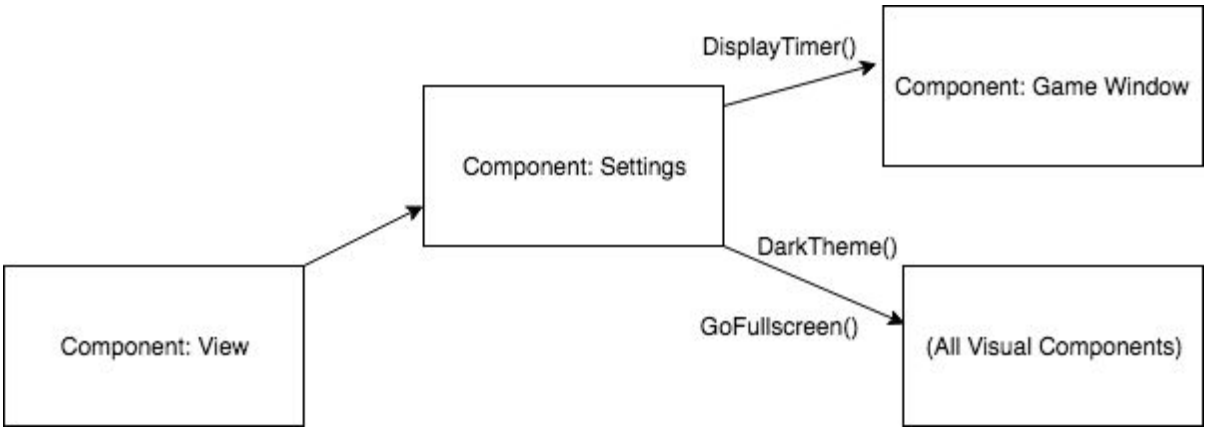
Inputs: Google sign in credentials
Outputs: Authentication guard unlocked
Dependent Components: Authenticate, Auth Route Guard



Settings Component

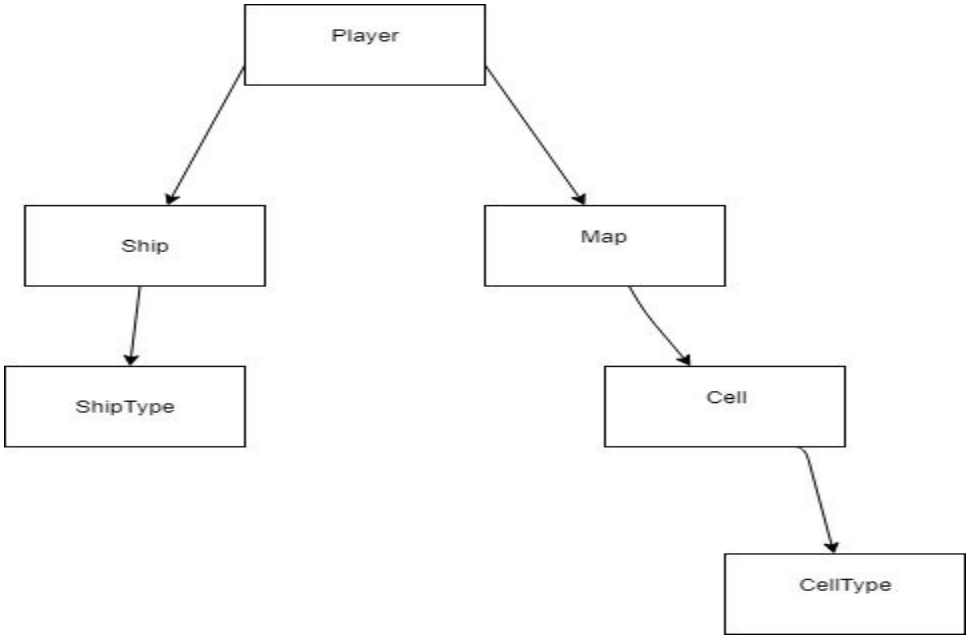
Outputs: DarkTheme [True/False], Fullscreen [T/F], Timer [T/F]

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



Incremental Testing Form

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.

Incremental Testing

Defect No.	Description	Severity	How to correct
1	(Main Menu) Application should render menu view on request of /home	1	Configured route ('/home') to respond with the Menu component in app.module.ts
2	(Main Menu) Application should render new game menu view on click of "New Game" button in menu view	1	Configure route ('/newgame') to respond with the menu component in app.module.ts. Upon button click, the application routes user to new game route (in newGameClicked() in menu.component.ts).
3	(Main Menu) Application should render new settings view on click of "Settings" button in menu view	1	Configure route ('/settings) to respond with the menu component in app.module.ts. Upon button click, the application routes user to settings route (in settingsClicked() in menu.component.ts).
4	(Main Menu)	1	Configure route ('/statistics) to

	Application should render new statistics view on click of “Statistics” button in menu view		respond with the menu component in app.module.ts. Upon button click, the application routes user to statistics route (in statisticsClicked() in menu.component.ts).
5	(Main Menu) Application should render new create strategy view on click of “Create strategy” button in menu view	1	Configure route (‘/createstrategy) to respond with the menu component in app.module.ts. Upon button click, the application routes user to create strategy route (in createStrategyClicked() in menu.component.ts).
6	(New Game Menu) Application should render the message “Place your carrier (5 Spaces)” when opened	3	Added a message element in the view and binded its contents to a {{message}} model that is managed by the program.
7	(New Game Menu) Application renders ships that have been placed already in the new game menu	1	Maintain a list of where ships have been placed
8	(New Game Menu) In new game menu, Application renders a grid for the user to interact with	1	Use of custom <app-grid> compnent that allows parent controller to register a onClick() listener
9	(New Game Menu) In new game menu, Application renders the form for enters a victory message and username	1	Created form component that is bounded to the {{victorymessage}} and {{username}} models respectively.
10	(Game Window) In the game window, Application renders two grids for user/enemy	1	Create two <app-grid> components for user and enemy
11	(Game Window) In the game window, Application should render action buttons attack ship, Move ship, surrender	1	Buttons were added to represent each action
12	(Game Window) In the the game window, Application should render a	1	Paragram element was added and bounded to {{message}} element in

	message component that informs the user what to do		game-window.component.ts
13	(Game Window) In the game window, Clicking a cell on either grid notifies user where in grid the cell was clicked (x and y)	1	Using element ID's, position is remembered for each cell in the grid angular component. A a cell click, cell pushes up its row, col and 1d index to any component who is listening on grid.
14	(Authentification) Application should render login view when visiting (/login)	1	Login route registered in app.module.ts on path /login
15	(Authentification) Application should render google sign in api button on login view	1	Implemented using Google Sign-In API.
16	(Settings) Switches should adjust state based on whether or not setting is active/inactive	1	Check state when loading and creating components; adjust switches accordingly
17	(Settings) Application should run in fullscreen mode when the user clicks the FS button	1	Link button to goFullScreen() function that will request fullscreen display based on browser being used
18	(Model) Cell should print "0" for water instead of "1" when checking for cell type.	1	Change ternary operator for checking cell types.
18	(Model) When a cell's type changes it must be handled one by one.	1	Edited for loop to traverse the grid of cells one by one.
19	(Model) When the ship is initialized, it should show the properties of the initialized ship.	1	Check that the ship class has a fully functioning constructor and initializes all variables properly.
20	(Model) When a ship is added to the grid, it must be placed in a set of open coordinates in the user's grid.	1	Check the bounds for the grid and check that the ship can fit within those bounds.
21	(Model)	1	Check to see that the correctly

	When a user attacks and hits a ship, the cell must be correctly updated.		identified cell's "hit" value is updated accordingly.
22	(Model) When a player makes a move that wins the game, the game should automatically end.	1	The server should handle the checks on whether a user has won or lost without the use of an unnecessary endpoint.

Regression Testing

Defect No.	Description	Severity	How to Correct
1	(new game menu) User placing <ship-name> ship of n spaces in Component B (Place ship) causes Component A (View) to prompt user "Place <ship name> (n Spaces)"	1	Handle logic that checks which ship is being placed and use a if/else-if/else sequence to determine which ship is being placed -> what to prompt the user.
2	(new game menu) User not being finished placing ships in Component B (Place ship) causes Component A (View) to now allow user to click create game button	2	Set element in .html file originally to disabled. Upon user completely entering their ship placement, button becomes enabled.
3	(new game menu) (User is placing a ship) User selects non-adjacent cell in component B (Add ship) causes component C (Placement validation) to return an invalid placement.	2	Using logic in new-game-menu.component.ts, check each entered point to ensure it is adjacent to a point in the ships cooridinates. If there is no places placed for a ship yet, this function always return true
4	(new game menu) (User is placing a ship) User selects a non-colinear cell (to rest of placed ship) in component B (Add ships causes component C (Placement validation) to return an invalid placement.	2	Using logic in new-game-menu.component.ts, check to ensure each point entered in the ship maintains its co-linear property. To do this, points are considered in pairs and their slopes are compared, expecting each slope to be the

			same for each pair.
5	(new game menu) (User is finished placing ships) On user being completely done placing ships and entered info for victory message/username, Component B (Add ships module) causes Component D (Ship Placement POST) to format a request to the server to send information about where the user placed ships.	1	Using Typescript interfaces, a GameRequest object was made to convert into JSON and sned to the server. This functionality is accomplished by adding a onSubmit() listener that sends the request once the user clicks it.
6	(new game menu) (User is finished placing ships) On user bei ng complete done placing ships and NOT entering info for victory message/username, Component E (Victory Message and Username form) causes component A (View) to notify the user that the field must be filled.	1	In onSubmit(), validation logic was added to make sure the {{victorymessage}} and {{username}} fields are set.
7	(new game menu) (After entering inputs for new game) Component C (Post ship placement) causes Component A (View) to push to Game Window view	1	Added a call to route to gamewindow after a user has entered all their inputs.
8	(Game Window) (After loading the Game Window view) Component B (GET request for user ship placements) causes Component A (View) to render the placed ships on the user grid	1	Using Angulars http module, and httprequest is made to a backend service that sends ship placement back
9	(Authentication Guard) When a user has not logged in/authentication, Component B (Login auth guard) causes Component C (any view component being requested) to reroute user	2	Using Route Activations, a logging guard was added to every route in the application to ensure no route can be visited without being authenticated. If the user is not authenticated, the application routes the user to the login page.
10	(Authentification)	2	After checking that the user has

	When a user successfully logs in using Google Signin Button in Component D (Authenticate) it causes Component A (View) to push to the main menu view		been authenticated in onSignIn() function, add a routing function to push user to main menu if the user was successfully authenticated.
11	(Settings) When the user decides to toggle on the “Dark Mode” option, the whole system display much change accordingly and remain in that state as long as the switch is set to True (All Visual Components)	1	Created global variable that each page will check on initialization to ensure proper color them is displayed
12	(Model) When the cell type was changed for an individual cell, the entire row of the respective cell was changed.	1	Edited the loop that handled the cell type changes and fixed the logic so it would properly traverse and configure individual cells in 2d arrays.