CLASSIFICATION OF COMPONENTS
Module description

Tic Tac Toe Module:

- Module is in charge of allowing user to play tic tac toe.
- The Tic Tac Toe activity component connects to the Tic Tac Toe java class. The java class provides all of the entire logic of the game and the activity contains the buttons and UI.
- Input: User presses buttons to place game piece
- Output: Game displays score, AI piece placement, and board

Connect 4 Module:

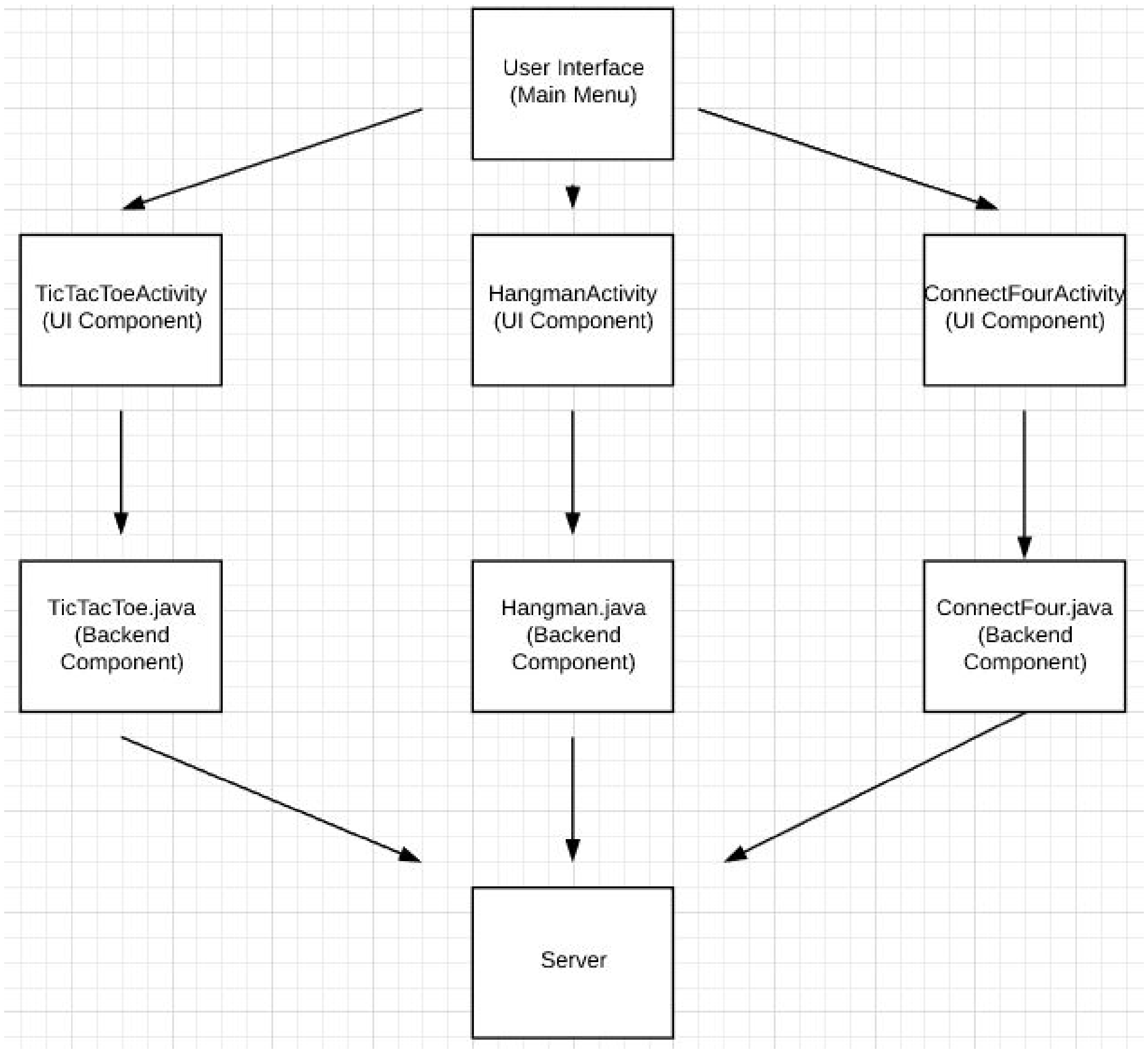
- Module is in charge of allowing user to play connect 4.
- The connect 4 activity component connects to the connect 4 java class. The java class provides all of the entire logic of the game and the activity contains the buttons and UI.
- Input: User selects column to place game piece
- Output: Game displays score, AI piece placement, and board

Hangman Module:

- Module is in charge of allowing user to play hangman.
- The hangman activity component connects to the hangman java class. The java class provides all of the entire logic of the game and the activity contains the buttons and UI.
- Input: User enters letter or an entire word
- Output: Game displays score, letters chosen, and number of wrong guesses

Incremental testing technique:

Testing approach is bottom up where the modules below were implemented and tested, integrated with modules above them and tested again till the interfaces reflected the changes/bug-fixes.


User Interface (Main Menu)
TicTacToeActivity (UI Component)
HangmanActivity (UI Component)
ConnectFourActivity (UI Component)
TicTacToe.java (Backend Component)
Hangman.java (Backend Component)
ConnectFour.java (Backend Component)
Server

Since we used the bottom up approach, we used drivers for the first phase of testing to verify the functionality of the modules. Later, upper level modules were added along while being tested.

Incremental and Regression Testing
Automation

Tic Tac Toe:

Tic Tac Toe testing uses the Java built-in junit library and the tests cover different types of inputs and edge cases. Furthermore, incremental and regression tests were created to test specific test cases and ensure fixing bugs did not create more bugs.

Connect 4:

Connect 4 testing uses the Java built-in junit library and the tests cover different types of inputs and edge cases. Furthermore, incremental and regression tests were created to test specific test cases and ensure fixing bugs did not create more bugs.

Hangman:

Hangman testing uses the Java built-in junit library and the tests cover different types of inputs and edge cases. Furthermore, incremental and regression tests were created to test specific test cases and ensure fixing bugs did not create more bugs.

Defect Log:

Module	Tic Tac Toe		
Defect No.	Description	Severity	Solution
1	newGame function was in an infinite loop which caused the game to hang	3	Remove infinite loop and call newGame function when a new game is created
2	When clicking a certain button on the screen when all other buttons were pressed, the game would crash	2	Fixed by reworking some of the logic and eliminated a race condition
3	CheckBlock function had an if statement which would check the	3	Fixed by debugging

	board out of bounds and cause the game to crash		and fixing the incorrect if statement

Module	Connect 4		
Defect No.	Description	Severity	Solution
1	In the check3Winner function the code for checking diagonally would go out of bounds when the pieces were placed at edges. This would cause the game to crash.	3	While debugging we found the line of code which caused this issue and added an out of bounds check to prevent the crash.
2	Vertical checks by AI would cause the AI to place pieces in already occupied spots.	3	Made sure to only attempt to place a piece if there was an empty spot
3	For horizontal checks in the check3Winner function, the AI wouldn’t recognize a 3 in a row if started from the right side of the board.	3	Added a check that would inspect the board from the right side as well.
4	Columns were not aligned properly in the UI which caused the app to look bad	1	Fixed by correctly calculating the width of the screen and using

			that to evenly space everything

Module	Hangman		
Defect No.	Description	Severity	Solution
1	Guessing the same letter twice would penalize the user instead	2	Display a message when the user presses the same button twice and not penalize the player
2	getRandomWord function would not read lines from the file correctly (certain lines were being missed)	3	Fixed by correctly keeping track of which line was read
3	getNumAppendages function did not reset the number of appendages when a new game was started	3	Fixed by resetting certain variables when a new game began.