Escort The President: Observation Testing

Team A1

Revision Log

Person	Action	Date
James Birch	Added tests and executed plan.	19th & 20th March 2017
Ahmed Bhallo	Wrote initial collection of tests.	18th March 2017

Action Required

Test ID	Problem	Fixed?
1.3.5	Join button should be greyed out when selecting a lobby that is in-game.	Yes
1.3.6	Join button should be greyed out when selecting a lobby that is full.	Yes
1.4.6	IllegalArgumentException thrown by server when trying to run a game when setting the map as 'University'.	Yes
1.6.1	Settings menu does not come up in game when pressing ESC on the keyboard.	Yes
1.6.5	Music and sound not sounding again after muting then unmuting from the settings menu in game.	Yes
1.6.5	Background music goes back to the start upon clicking 'Apply changes'.	No
1.6.6	Notify the user of the success of applying changes in the settings menu.	No
1.6.7	Game does not play with default key bindings when changes are made to them in the settings then 'Restore defaults' is applied.	Yes
3.7.4	Reload sound not played.	Yes
3.7.5	Power-up sound not played.	Yes

Contents

1 UI Menus

1.1 Server Connection Menu

Test ID:	1.1.1
Name:	User notified when invalid input is entered.
Description:	When the user leaves empty fields, or inputs illegal characters, appropriate error messages must be displayed. The error message must stand out and be highly visible.
Steps to reproduce:	1 Leave any of the 3 fields blank 2 Click on the connect button.
Expected result:	The status text is updated to reflect the error made by the user. Status text is a bright contrasting colour.
Actual result:	The connection is refused and the status text shows the message "You must fill in all fields". This is in the colour red which contrasts with the background well.
Action:	None.

Test ID:	1.1.2
Name:	User notified when desired username is in use.
Description:	When the user attempts to connect to the server, but there is another client with the same username already connected, the connection is reject and the user is notified.
Steps to reproduce:	1 Start two clients and one server instances. 2 Enter "user" as the username on one of the clients and connect to the server. 3 On the other client, enter the same username ("user"), and attempt to connect.
Expected result:	The connection should be rejected and the status text is updated to notify the client that their desired username is already in use.
Actual result:	The connection is refused and the status text shows the message "That name is currently already in use".
Action:	None.

Test ID:	1.1.3
Name:	Username is freed up again after the user leaves.
Description:	The username that a user takes is freed up after they leave the server.
Steps to reproduce:	1 Start two clients and one server instances. 2 Enter ”user” as the username on one of the clients and connect to the server. 3 On the other client, enter the same username (”user”), and attempt to connect. 4 Then, the first client disconnects from the server. 5 The second client enters the same username again (”user”), and attempts to connect.
Expected result:	The connection should be rejected to the second client while the first client is still on the server, then the server should accept the second client after the first one disconnects.
Actual result:	The second client is rejected while the first client is connected to the server but then the second client is accepted after the first client disconnects.
Action:	None.

Test ID:	1.1.4
Name:	User notified when desired username is too long or contains non-alphanumeric characters.
Description:	When the user attempts to connect to the server with a username that is greater than 12 characters long, the connection is rejected and the user is notified. The validation must be done client-side as well as server-side in order to reduce server load.
Steps to reproduce:	1 Start a client. 2 Enter ”user123456789123456” and ”user;” as the username (or any other username with length greater than 12 characters or contains non-alphanumerics) on one of the clients. 3 Click the connect button.
Expected result:	The connection to the server is not attempted and the user is notified that their desired username is too long.
Actual result:	The connection is not attempted and the status text ”Names must be alphanumeric and at most 12 characters” is displayed for both inputs.
Action:	None.

Test ID:	1.1.5
Name:	User enters invalid port number.
Description:	When the user attempts to connect to a server with text inside the port number input field that cannot be parsed as a decimal number, the connection is rejected and the user is notified.
Steps to reproduce:	1 Start a client. 2 Enter ”abc” into the server port field 3 Click the connect button.
Expected result:	The connection to the server is not attempted and the user is notified that their entered port number is not a valid number.
Actual result:	The connection is not attempted and the status text ”Invalid port number” is displayed.
Action:	None.

Test ID:	1.1.6
Name:	User notified when connection to server cannot be made.
Description:	When the user attempts to connect to a server, but the connection cannot be made (lack of a stable internet connection or typo in server name/port), the user is notified.
Steps to reproduce:	1 Start a client. 2 Enter an invalid server name or disconnect your internet connection. 3 Click the connect button.
Expected result:	The user is notified that the connection to the server cannot be established.
Actual result:	The connection is not successfully made and the status text ”Could not establish a connection to that server” is displayed.
Action:	None.

Test ID:	1.1.7
Name:	Server does not have a certificate signed by a trusted authority.
Description:	When the user attempts to connect to a server that does not have a trusted certificate, a prompt is made to ask the user to manually vertify the public signature of the server.
Steps to reproduce:	1 Start a client. 2 Enter the details of a server that does not have a trusted certificate. 3 Click the connect button.
Expected result:	The user is notified that the server does not have a trusted certificate. A dialog is presented with the server's public key. The user is asked to accept or reject the connection to the server.
Actual result:	The user is notified with a dialog box that they need to trust the connection manually. Cancelling takes the user back to the front screen and does establish the connection whereas clicking 'I trust this connection' successfully creates the connection and takes the user to the Main Menu.
Action:	None.

1.2 Main Menu

Test ID:	1.2.1
Name:	User clicks on a menu button and is redirected to the appropriate menu.
Description:	When the user clicks on a menu button, they should be taken to the appropriate menu that they want to go to.
Steps to reproduce:	1 Start a client and connect to server. 2 Click on a menu button.
Expected result:	The user should now be redirected to the appropriate menu.
Actual result:	1 Clicking the 'Lobby list' button successfully takes the user to the lobby list menu. 2 Clicking the 'How to play' button successfully takes the user to the how to play pages. 3 Clicking the 'Settings' button successfully takes the user to the settings menu.
Action:	None.

Test ID:	1.2.2
Name:	User clicks on the 'Disconnect from server".
Description:	Disconnecting a user from the server.
Steps to reproduce:	1 Start a client and connect to a server. 2 Click 'Disconnect from server'.'
Expected result:	A dialog box is produced telling the user they have been disconnected and the user is disconnected from the server. Selecting 'OK' should take the user back to the front screen.
Actual result:	A dialog box does appear and the server is disconnected. Selecting 'OK' takes the user back to the front screen.
Action:	None.

Test ID:	1.2.3
Name:	User clicks on exit button. Confirmation dialog is displayed.
Description:	When the user clicks on the exit menu button, a warning message should pop up asking the client to confirm that they want to leave. This is to prevent accidental exit of the client.
Steps to reproduce:	1 Start a client and connect to server. 2 Click on the exit menu button.
Expected result:	A warning dialog is displayed and the client is asked to confirm if they really want to close the client. If no is selected, the dialog is removed. If yes is selected, the client closes.
Actual result:	A warning dialog does get displayed. Clicking 'Cancel' takes the user back to the main menu successfully and clicking 'Quit' does successfully close the client.
Action:	None.

1.3 Lobby List Menu

Test ID:	1.3.1
Name:	List of all lobbies are displayed.
Description:	When the user enters the lobby list menu, the list of all lobbies are shown and automatically refreshed every set amount of seconds. For each lobby, its name, information on the amount of players, in-game status, whether or not it is locked, and owner and displayed.
Steps to reproduce:	1 Connect to a server. 2 From the main menu, click on the lobby list button.
Expected result:	All lobbies on the server are displayed with their appropriate information. Every few seconds this list is automatically refreshed.
Actual result:	The lobbies on the server are successfully displayed. They update with the appropriate information (in game and locked) successfully when the lobby is in game and/or locked respectively.
Action:	None.

Test ID:	1.3.2
Name:	Lobby list filtering applied when selected
Description:	When the user selects a filter on the lobby list menu, all filters are then applied to the lobby lists.
Steps to reproduce:	1 Connect to a server. 2 From the main menu, click on the lobby list button. 3 Click on any combination of filtering.
Expected result:	The list of lobbies should be filtered on the user's choosing. For example, if the user selects to only see unlocked games that are not already in game, only these lobbies should be displayed in the list.
Actual result:	1 Filtering lobbies on whether they are full hides appropriately successfully. 2 Filtering lobbies on whether they are locked (password-protected) hides appropriately successfully. 3 Filtering lobbies on whether they are in game hides appropriately successfully.
Action:	None.

Test ID:	1.3.3
Name:	Create a lobby successfully.
Description:	The user should be able to create a lobby (with or without a password).
Steps to reproduce:	1 Have a client connect to a server and go to the lobby list menu. 2 Click on ’Create new’. 3 Enter a lobby name (and a password if desired). 4 Click on ’Create’.
Expected result:	The lobby should be created and viewable to other clients.
Actual result:	The lobby is created and vieable by other clients. This works for both lobbies created with and without a password.
Action:	None.

Test ID:	1.3.4
Name:	Join a lobby successfully.
Description:	The user is able to join unlocked lobbies freely and locked lobbies by supplying the correct password.
Steps to reproduce:	1 Have two clients connected to a server instance. 2 Get the first client to create an unlocked lobby. 3 On the second client, click on the unlocked lobby and click ’Join’. 4 Now, get the first client to leave and create a new locked lobby with any password. 5 Then get the second client to leave the first (unlocked) lobby and then click the new locked lobby and click ’Join’. The second client should then supply the correct password.
Expected result:	The second client should be able to join both the unlocked lobby and the locked lobby successfully (as separate actions).
Actual result:	The second client is able to join both lobbies successfully.
Action:	None.

Test ID:	1.3.5
Name:	Refuse joining a lobby if it is in game.
Description:	The user should not be able to join a lobby that is in a game.
Steps to reproduce:	1 Have two clients connected to a server instance and be in the lobby list menu. 2 Get the first client to create a lobby and start a game. 3 On the second client, click on the lobby which is in-game and click 'Join'.
Expected result:	The second client should be refused access.
Actual result:	The second client seems to be refused access but there is no notification to the user of this refusal. Stderr shows the message 'Unimplemented message: 29'.
Action:	The 'Join' button should be greyed out when the lobby is in-game.

Test ID:	1.3.6
Name:	Refuse joining a lobby if it is full.
Description:	The user should not be able to join a lobby that is full. Take the limit to be n clients (displayed on the lobby list menu).
Steps to reproduce:	1 Have n+1 clients connected to a server instance and be in the lobby list menu. 2 Get the first client to create a lobby and get another n-1 of the clients to join. 3 On the n+1th client, click on the lobby and click 'Join'.
Expected result:	The n+1th client should be refused access.
Actual result:	The n+1th client seems to be refused access but there is no notification to the user of this refusal. Stderr shows the message 'Unimplemented message: 30'.
Action:	The 'Join' button should be greyed out when the lobby is full.

Test ID:	1.3.7
Name:	Refuse if incorrect password is given to a password-protected lobby.
Description:	The user should not be prevented from joining a password-protected lobby if they provide the wrong password.
Steps to reproduce:	1 Have a two clients connected to a server instance and have them both on the lobby list menu. 2 Get the first client to create a password-protected lobby. 3 Get the second client to try and join the first client's lobby by clicking on their lobby then clicking 'Join'. 4 The second client should then enter the wrong password and click 'Join'.
Expected result:	The second client is refused access and shown a dialog box informing them of this.
Actual result:	The second client is refused access and a dialog box is shown. Clicking 'Ok' 'takes the user back to the lobby list menu.
Action:	None.

Test ID:	1.3.8
Name:	Lobby refreshes when a new lobby is created on the server.
Description:	The user should be able to see in real-time if a new lobby has been created on the server.
Steps to reproduce:	1 Have a two clients connected to a server instance and have them both on the lobby list menu. 2 Get the first client to create a lobby. 3 The second client should be able to see that lobby.
Expected result:	The second client can see the newly created lobby on their lobby list.
Actual result:	The second client can successfully see the first client's lobby.
Action:	None.

Test ID:	1.3.9
Name:	Go back button on the lobby list menu.
Description:	The user clicks on the ’Go back’ button.
Steps to reproduce:	1 Have a client connected to a server instance and on the lobby list menu. 2 Click the ’Go back’ button.
Expected result:	The user should be returned to the game’s main menu.
Actual result:	The user is successfully taken back to the main menu.
Action:	None.

1.4 Game Lobby

Test ID:	1.4.1
Name:	Access how to play pages from game lobby.
Description:	The user should be able to click on ’Click here to view how to play’ to view the How To Play pages.
Steps to reproduce:	1 Have a client connected to a server instance and in a game lobby. 2 Click ’Click here to view how to play’.
Expected result:	The user is taken to the How To Play pages.
Actual result:	The user is successfully taken to the How To Play pages.
Action:	None.

Test ID:	1.4.2
Name:	Access settings menu from game lobby.
Description:	The user should be able to click on ’Open settings’ to view the settings from the game lobby.
Steps to reproduce:	1 Have a client connected to a server instance and in a game lobby. 2 Click ’Open settings’.
Expected result:	The user is taken to the settings menu.
Actual result:	The user is successfully taken to the settings menu.
Action:	None.

Test ID:	1.4.3
Name:	Enter chat in lobby.
Description:	The user should be able to enter chat in a game lobby and it should appear on all clients in a lobby.
Steps to reproduce:	1 Have two clients connected to a server instance and in the same game lobby. 2 On one of the clients press 'Enter' and then type a message. Press 'Enter' to send it.
Expected result:	The message should be viewable on all clients in the game lobby.
Actual result:	The message is successfully viewable on all clients.
Action:	None.

Test ID:	1.4.4
Name:	Change settings then return to game lobby.
Description:	The user should be able to return to the game lobby after changing their settings. The state of the lobby should be unchanged after return. The 'Go back' button in the Settings menu is also tested here.
Steps to reproduce:	1 Have a client connected to a server instance and in a game lobby. 2 Enter some random chat by pressing 'Enter' followed by your message followed by 'Enter' again. 3 Click 'Open settings'. 4 Change the resolution and click 'Apply changes' then 'Keep'. 5 Click 'Go back' on the settings menu.
Expected result:	The user should be returned to the lobby and it should be the same (players nor chat messages should disappear).
Actual result:	The user is successfully returned and the lobby is the same as when left.
Action:	None.

Test ID:	1.4.5
Name:	Kick player from a lobby.
Description:	The owner of a lobby should be able to kick other players (clients that are not the owner should not be able to kick).
Steps to reproduce:	1 Have two clients connected to a server instance and in the same game lobby. 2 The client who is the lobby owner should click 'Kick' next to the other client's name.
Expected result:	The client who received the kick should be removed from the lobby. The other client should not have seen any option to kick while they were in the lobby.
Actual result:	The user who received the kick is removed from the lobby. Also, the other client had no option to kick anyone else.
Action:	None.

Test ID:	1.4.6
Name:	Start game.
Description:	The owner of a lobby should be able to start the game (other clients should not be able to start the game).
Steps to reproduce:	1 Have two clients connected to a server instance and in a game lobby. 2 The client who is the lobby owner should click 'Start game'. The other client should not be able to change these values. 3 The owner should then start the game.
Expected result:	The game should start up.
Actual result:	The game starts.
Action:	None.

Test ID:	1.4.7
Name:	Change AI values.
Description:	The owner of a lobby should be able to change the AI values and the map.
Steps to reproduce:	1 Have two clients connected to a server instance and in a game lobby. 2 The client who is the lobby owner should change the values of the AI so that there is 1 of each AI type (for counting purposes) and make the map ’Hotel’. The other client should not be able to change these values. 3 The owner should then start the game. 4 Repeat the above for 2 of each AI and setting the map to University.
Expected result:	Couting the number of AI in the game should equal what was set. The other client should not be able to change these values
Actual result:	The AI count is correct. Also, the other client had greyed out boxes next to the counts so they could not change it. The server threw an IllegalArgumentException when trying to run the game with the map set as University.
Action:	Inspect the cause of the IllegalArgumentException when running the University map.

Test ID:	1.4.8
Name:	Go back.
Description:	The user should be able to go back to the lobby list menu (effectively leaving the lobby).
Steps to reproduce:	1 Have a client connected to a server instance and in a game lobby. 2 The user should click the ’Go back’ button.
Expected result:	A dialog box should pop up informing the user they are about to leave the lobby. Clicking ’Cancel’ should keep them in the lobby and clicking ’Yes’ should cause them to go back to the lobby list menu.
Actual result:	A dialog box is successfully produced. Clicking ’Cancel’ keeps the user in the lobby and clicking ’Yes’ does cause the user to leave the lobby.
Action:	None.

1.5 How To Play Menu

Test ID:	1.5.1
Name:	Switch pages.
Description:	The user should be able to switch left and right between the help pages.
Steps to reproduce:	1 Have a client connected to a server instance and on the 'How to play' pages. 2 The user should click any sequence of the left arrow and right arrow on screen to navigate pages.
Expected result:	The page should change according to whether left or right is pressed and should be consistent in its changes. Going as far left as possible should result in the left arrow being greyed out and going as far right as possible should result in the right arrow being greyed out.
Actual result:	The pages do change correctly and consistently. The buttons also grey out at the expected times (left arrow on leftmost page and right arrow on rightmost page).
Action:	None.

Test ID:	1.5.2
Name:	Text on different resolutions.
Description:	The text on the help pages is legible and spaced consistently on different resolutions.
Steps to reproduce:	1 Have a client connected to a server instance and on the 'How to play' pages. 2 View the text. 3 Open a new client and connect it a server instance and go to the settings menu. Change the resolution and click 'Apply changes'. 4 Go back to the main menu by clicking 'Go back' and then click on 'How to play'. 5 View the text.
Expected result:	The text should be consistent and legible.
Actual result:	The text is legible and spaced consistently on all resolutions.
Action:	None.

Test ID:	1.5.3
Name:	Go back.
Description:	The user should be able to go back to the main menu.
Steps to reproduce:	1 Have a client connected to a server instance and on the 'How to play' pages. 2 The user should click the 'Go back' button.
Expected result:	The user is returned to the main menu.
Actual result:	The user is successfully returned to the main menu.
Action:	None.

1.6 Settings Menu

Test ID:	1.6.1
Name:	Settings can be accessed in game.
Description:	The user should be able to access the settings while in game (by clicking on the settings button or by pressing ESC on the keyboard).
Steps to reproduce:	1 Have a client connected to a server instance and in a game. 2 The user should click on the 'Settings' button in the top right (or press ESC on the keyboard).
Expected result:	The settings menu should be brought up in game.
Actual result:	The settings menu does come up when clicking on the button but not when pressing ESC.
Action:	Investigate why the settings menu does not come up when pressing ESC.

Test ID:	1.6.2
Name:	Changing the client's resolution.
Description:	The user should be able to change the client's resolution to match their machine's capabilities. The user should be able to revert the changes if they are not happy with the change.
Steps to reproduce:	1 Have a client connected to a server instance and in the settings menu. 2 The user should change the resolution. 3 The user should click 'Apply changes' and then click 'Revert'. 4 The user should click 'Apply changes' and then let the timer run out. 5 The user should click 'Apply changes' and then click 'Keep'.
Expected result:	The resolution should change to the new resolution if 'Keep' is pressed otherwise clicking 'Revert' (or letting the timer run out) changes the resolution back.
Actual result:	The resolution is successfully changed and kept when clicking 'Keep' but clicking 'Revert' or letting the timer run out successfully changes it back.
Action:	Investigate why the settings menu does not come up when pressing ESC.

Test ID:	1.6.3
Name:	Detect resolution.
Description:	The client should be able to detect the machine's resolution and give that as a suggestion for the user to apply.
Steps to reproduce:	1 Have a client connected to a server instance and in the settings menu. 2 The user clicks on 'Detect resolution'.
Expected result:	The resolution should be changed in the options (but not applied unless 'Apply changes' is selected).
Actual result:	The resolution is changed in the options based on the machine's capabilities.
Action:	None.

Test ID:	1.6.4
Name:	Change music and sound levels.
Description:	The user should be able to change the game's music and sound levels from the settings menu while in game.
Steps to reproduce:	1 Have a client connected to a server instance and in a game. 2 The user clicks on 'Settings' in the top right. 3 The user changes the music and sound levels.
Expected result:	The music and sound levels should change according to how the user changes them (Higher percentage is louder).
Actual result:	The music and sound levels do increase and decrease as expected from the changes made in the settings menu. The music and sound do both go silent when 0% is selected which is good.
Action:	None.

Test ID:	1.6.5
Name:	Mute and unmute music and sound.
Description:	The user should be able to mute and unmute the game's music and sound levels from the settings menu while in game.
Steps to reproduce:	1 Have a client connected to a server instance and in a game. 2 The user clicks on 'Settings' in the top right. 3 The user mutes the music and sound levels and clicks 'Apply changes'. 4 The user unmutes the music and sound levels and clicks 'Apply changes'.
Expected result:	The music and sound should be silent while mute is selected and loud again when mute is not selected.
Actual result:	The music and sound levels do go silent when mute is applied but when unmuting the music and sound do not start playing again. Also, clicking 'Apply changes' seems to restart the background music.
Action:	Investigate why the music is not sounding again when unmuting the music and sound is applied. Also, need to stop background music from being restarted by clicking on 'Apply changes'.

Test ID:	1.6.6
Name:	Change key bindings.
Description:	The user should be able to change the key bindings and they should carry through into game.
Steps to reproduce:	1 Have a client connected to a server instance and in a game. 2 The user clicks on ’Settings’ in the top right. 3 The user changes some of their key bindings. 4 The user clicks ’Apply changes’.
Expected result:	The bindings changed are changed in game. The user should be notified of the success of the application of the changes.
Actual result:	The key bindings are successfully changed in game. ’Apply changes’ does not notify the user that changes were completed successfully.
Action:	Have a message that notifies the user of the success of the application of the changes.

Test ID:	1.6.7
Name:	Restore default key bindings.
Description:	The user should be able to restore their key bindings to the game’s set defaults.
Steps to reproduce:	1 Have a client connected to a server instance and in the settings menu. 2 The user changes their key bindings. 3 The user clicks ’Apply changes’. 4 The user clicks ’Restore defaults’. 5 Click ’Go back’, then create a lobby and start the game.
Expected result:	The bindings should be changed back to the game’s defaults. Playing the game should verify this.
Actual result:	The key bindings change back to the game’s defaults visually in the settings menu but the game plays using the changes that were previously made (and not the defaults).
Action:	Investigate why the default key bindings are not being used in game.

Test ID:	1.6.8
Name:	Discard changes to key bindings.
Description:	The user should be able to discard their changes to the key bindings in the settings menu.
Steps to reproduce:	1 Have a client connected to a server instance and in the settings menu. 2 The user changes their key bindings. 3 The user clicks 'Discard changes'. 4 Click 'Go back', then create a lobby and start the game.
Expected result:	The bindings should be changed back to what they were when they were last applied. Playing the game should verify this.
Actual result:	The key bindings are successfully changed back to what they were when they were last applied. Playing the game is done successfully with what the controls were previously.
Action:	None.

Test ID:	1.6.8
Name:	Go back.
Description:	The user should be able to go back to the main menu from the settings menu in the menu structure.
Steps to reproduce:	1 Have a client connected to a server instance and in the settings menu. 2 Click 'Go back'.
Expected result:	A dialog box asks if the user understands that any unsaved changes will be lost. Clicking 'I understand' should take the user back to the main menu whereas clicking 'Cancel' should keep the user on the settings menu.
Actual result:	A dialog box successfully pops up asking if the user understands that any unsaved changes will be lost. Clicking 'I understand' does take the user back to the main menu and clicking 'Cancel' does not take the user away from the settings menu.
Action:	None.

2 UI Components

2.1 Input Field

Test ID:	2.1.1
Name:	Input fields change colour when focussed.
Description:	When the user clicks onto an input field that is not disabled, the background and border colour should both change. It should be clear to the user which input field they are currently typing into. The user should not accidentally type into the wrong field.
Steps to reproduce:	1 Hover mouse of a non-disabled input field. 2 Perform a left mouse click action.
Expected result:	The border of the input field should now be a bright contrasting colour. The background of the input field should now be brighter than when it was not focussed.
Actual result:	The input field is brighter when in focus and the border contrasts using the colour light blue which works well.
Action:	None.

Test ID:	2.1.2
Name:	Focussed input fields become unfocussed when the mouse is clicked elsewhere.
Description:	When a input field is focussed, and the user clicks onto an area of the client that is not inside the input field, the input field should become unfocussed. Only one input field may be focussed at any one time.
Steps to reproduce:	1 Ensure a input field is already focussed. 2 Move the mouse to an area that is not on the focussed input field. 3 Perform a left mouse click action.
Expected result:	The input field becomes unfocussed; it should not detect any key-presses and its colours should change accordingly.
Actual result:	The input field successfully becomes unfocussed. No key presses are detected by the input field when in this state.
Action:	None.

Test ID:	2.1.3
Name:	Input field focus should be traversed when TAB key is pressed.
Description:	When a input field is focussed and the user presses the TAB key, the focus of the input field should be traversed to the next specified input field (if it exists).
Steps to reproduce:	1 Ensure a input field is already focussed. 2 Press the TAB key.
Expected result:	If the previous input field had a specified next input field, this new input field is now focussed and the old one is unfocussed.
Actual result:	Pressing TAB successfully traverses the input fields in a logical way, leaving the previous input field unfocussed and the new input field focussed.
Action:	

Test ID:	2.1.3
Name:	The caret position of a focussed input field is moved according to the mouse press.
Description:	When a input field is focussed and the user clicks on text inside it, the caret position should change according to where the mouse is.
Steps to reproduce:	1 Click on an input field so that it is focussed. 2 Enter some text. 3 Click/drag across the text.
Expected result:	The position of the caret should move according to which character the mouse is over when the click was registered.
Actual result:	The position of the caret is successfully moved according to where the mouse is pressed/dragged.
Action:	None.

2.2 Check Boxes

Test ID:	2.2.1
Name:	Highlighting check boxes.
Description:	Check boxes highlight when they are selected and unhighlight when they are selected again. These check boxes can be found on the lobby.
Steps to reproduce:	1 Click on the check boxes in a lobby. 2 Click on the same check boxes again.
Expected result:	The check boxes is highlighted when clicked the first time and then the check box unhighlights when selected again.
Actual result:	The check boxes successfully highlight and unhighlight.
Action:	None.

3 Gameplay

3.1 Health Bars

Test ID:	3.1.1
Name:	Health bars full at start of game.
Description:	At the start of a game, the health bars of all units should be full.
Steps to reproduce:	1 Have a client in game.
Expected result:	The health bars of all units should be initially full.
Actual result:	The health bars of all units are all full at the start.
Action:	None.

Test ID:	3.1.2
Name:	Health bars deplete when taking damage.
Description:	When a unit takes damage their health should deplete.
Steps to reproduce:	1 Have a client in game. 2 The user shoots at an enemy. 3 The user throws a grenade at an enemy.
Expected result:	If any of the bullets hit the unit or the unit is range of a grenade explosion then their health bar should be depleted.
Actual result:	The health bar of the unit successfully depletes when shot or caught in the range of an exploding grenade.
Action:	None.

3.2 Shooting, Weapons, and Shields

Test ID:	3.2.1
Name:	Shooting fires a visible bullet.
Description:	When a user shoots, a bullet should be seen on the screen.
Steps to reproduce:	1 Have a client in game. 2 The user shoots the pistol. 3 The user switches to the machine gun and shoots that.
Expected result:	The pistol and machine gun should spawn visible bullets when shot.
Actual result:	The pistol and machine gun successfully spawns visible bullets when shot.
Action:	None.

Test ID:	3.2.2
Name:	Shooting decreases bullet count.
Description:	When a user shoots, the bullet count of the gun decreases.
Steps to reproduce:	1 Have a client in game. 2 The user shoots the pistol twice. 3 The user switches to the machine gun and shoots that 5 times.
Expected result:	The pistol and machine gun should decreases their bullet counts when shot depending how many bullets were shot. (-2 for pistol and -5 for machine gun).
Actual result:	The pistol and machine gun successfully deplete their bullet counts when shot (the pistol -2 and the machine gun -5).
Action:	None.

Test ID:	3.2.3
Name:	Bullets stop after hitting a unit.
Description:	When a user shoots, the bullet stops after hitting a unit or an obstacle (like a wall).
Steps to reproduce:	1 Have a client in game. 2 The user shoots the pistol towards a unit. 3 The user shoots the pistol towards a wall. 4 Repeat the above 2 steps for the machine gun.
Expected result:	The pistol and machine gun bullets should stop after making impact with a unit or obstacle.
Actual result:	The pistol and machine gun bullets successfully stop after hitting a unit or obstacle.
Action:	None.

Test ID:	3.2.4
Name:	Guns switch on the unit sprite when switching weapons.
Description:	When a user switches their weapon, the sprite changes the gun they are holding.
Steps to reproduce:	1 Have a client in game. 2 The user switches to the machine gun.
Expected result:	The gun on the sprite changes from the machine gun to the pistol.
Actual result:	The gun on the sprite successfully changes from the machine gun to the pistol.
Action:	None.

Test ID:	3.2.5
Name:	Shield is shown when selected.
Description:	When a user switches to the shield, it is shown as a yellow block on the unit's body with a white border.
Steps to reproduce:	1 Have a client in game. 2 The user switches to the shield.
Expected result:	The shield is displayed on the unit.
Actual result:	The shield is successfully displayed on the unit.
Action:	None.

Test ID:	3.2.6
Name:	Shield health depletes when taking damage.
Description:	The shield health decreases when taking damage.
Steps to reproduce:	1 Have two clients in a game 2 The first client switches to the shield. 3 The second client shoots the shield and then throws a grenade at the first client.
Expected result:	The shield health should deplete when selected and shot and when the unit is within the range of a grenade exploding.
Actual result:	The shield health is successfully depleted when shot and in the range of an exploded grenade (when selected by the user).
Action:	None.

Test ID:	3.2.7
Name:	Shield disappears when it has no health.
Description:	The shield disappears when it has no health.
Steps to reproduce:	1 Have two clients in a game 2 The first client switches to the shield. 3 The second client shoots the shield until it has no health.
Expected result:	The shield changes when the health of it is zero to be a transparent white border (exposing the unit's body).
Actual result:	The shield successfully changes when the health of the shield is zero (to the transparent white border which exposes the unit's body).
Action:	None.

Test ID:	3.2.8
Name:	Shield replenishes when a unit respawns.
Description:	The shield replenishes back to full health on respawn.
Steps to reproduce:	1 Have two clients in a game 2 The first client switches to the shield. 3 The second client shoots the shield until it has no health and then proceeds to kill the first client's unit. 4 Wait for first client to respawn.
Expected result:	The shield should replenish back to full when the user respawns.
Actual result:	The shield successfully replenishes its health when the user respawns.
Action:	None.

Test ID:	3.2.9
Name:	The weapon slot highlighted reflects the currently chosen weapon.
Description:	The weapon slots on screen highlight according to which weapon was desired.
Steps to reproduce:	1 Have a client in a game. 2 Switch to the machine gun. 3 Switch to the shield. 4 Switch to the pistol.
Expected result:	The weapon slot should highlight according to which weapon/shield is selected.
Actual result:	The weapon slot highlights correctly when each weapon/shield is selected.
Action:	None.

Test ID:	3.2.10
Name:	Reload bar above unit.
Description:	When a unit reloads, a progress bar fills up above the unit's head according to how much they have reloaded.
Steps to reproduce:	1 Have a client in a game. 2 The user shoots. 3 The user presses the reload key.
Expected result:	The reload bar should show above the unit's head and fill up with time according to the reload time of the weapon.
Actual result:	The reload bar successfully appears above the unit's head and fills up with time according to the reload time of the weapon.
Action:	None.

3.3 Grenades

Test ID:	3.3.1
Name:	Grenades are shown when held and thrown.
Description:	A grenade is displayed on screen when a unit holds a grenade and also when it is thrown.
Steps to reproduce:	1 Have a client in a game. 2 The user holds a grenade for 2 seconds. 3 The user releases the grenade key to throw it.
Expected result:	The grenade is shown when it is held and is shown to move when thrown.
Actual result:	The grenade is successfully shown when held and thrown.
Action:	None.

Test ID:	3.3.2
Name:	Grenades rebound off obstacles.
Description:	When a user throws a grenade that hits an obstacle, the grenade rebounds off the obstacle.
Steps to reproduce:	1 Have a client in a game. 2 The user presses the grenade key to throw a grenade and aims at a wall.
Expected result:	The grenade rebounds off the wall in a sensible manner.
Actual result:	The grenade successfully rebounds off the wall and does so in a sensible manner.
Action:	None.

Test ID:	3.3.3
Name:	Grenade animation on explosion.
Description:	When a grenade explodes, an animation is played.
Steps to reproduce:	1 Have a client in a game. 2 The user presses the grenade key to throw a grenade. 3 Wait until grenade explodes.
Expected result:	The grenade should play an explosion animation when exploding.
Actual result:	The grenade successfully plays an explosion animation.
Action:	None.

3.4 Power-ups

Test ID:	3.4.1
Name:	Power-up disappears after walking over it.
Description:	A power-up disappears after it is walked over to simulate it being picked up.
Steps to reproduce:	1 Have a client in a game. 2 The user walks over a power-up.
Expected result:	The power-up disappears if they are not maxed in that statistic.
Actual result:	The power-up successfully disappears if they are not maxed in that statistic (if they are maxed then it remains able to be collected).
Action:	None.

Test ID:	3.4.2
Name:	Power-up reappears after being taken.
Description:	A power-up reappears after a short time of disappearing due to being taken by a unit.
Steps to reproduce:	1 Have a client in a game. 2 The user walks over a power-up. 3 Wait for power-up to reappear.
Expected result:	The power-up should reappear and be able to be collected again.
Actual result:	The power-up successfully reappears and is able to be collected again.
Action:	None.

Test ID:	3.4.3
Name:	Health power-up increases health bar.
Description:	If a unit picks up a health power-up it adds health to their health bar for that unit.
Steps to reproduce:	1 Have a client in a game. 2 The user walks over a health power-up.
Expected result:	The health bar should increase when picking up the health power-up. It should not extend beyond its cap.
Actual result:	The power-up is successfully applied.
Action:	None.

Test ID:	3.4.4
Name:	Extra machine gun ammo power-up increases machine gun count.
Description:	If a unit picks up an extra machine gun ammo power-up it adds to the bullet counter for the machine gun for that unit.
Steps to reproduce:	1 Have a client in a game. 2 The user walks over an extra machine gun ammo power-up.
Expected result:	The machine gun bullet count should increase. It should not extend beyond its cap.
Actual result:	The power-up is successfully applied.
Action:	None.

Test ID:	3.4.5
Name:	Extra grenades power-up increases grenades count.
Description:	If a unit picks up an extra grenades power-up it adds to the grenade counter for that unit.
Steps to reproduce:	1 Have a client in a game. 2 The user walks over an extra grenades power-up.
Expected result:	The grenade count should increase. It should not extend beyond its cap.
Actual result:	The power-up is successfully applied.
Action:	None.

3.5 Units

Test ID:	3.5.1
Name:	Correct unit sprite numbers shown.
Description:	The correct number of units is shown in a game. This should be the sum of all the AI plus the number of players. There should be the number of players divided by 3 rounded up escorts, the number of players subtract the number of escorts as the number of assassins for players.
Steps to reproduce:	1 Have 3 client in a game. This means there should be 1 human escort and 2 human assassins. Set there to be 3 civilians, police, and assassins plus a president AI.
Expected result:	There should be 1 escort, 5 assassins, 3 civilians, 3 police and 1 president sprites in game.
Actual result:	The correct number of sprites is shown.
Action:	None.

Test ID:	3.5.2
Name:	Escorts, assassins and police carry guns.
Description:	The sprites for escorts, assassins and police all carry guns.
Steps to reproduce:	1 Have a client in a game with assassin and police AIs.
Expected result:	The sprites for escorts, assassins and police should carry guns.
Actual result:	The sprites do carry guns.
Action:	None.

Test ID:	3.5.3
Name:	Units are responsive to input.
Description:	The units are responsive to inputs from the mouse and keyboard.
Steps to reproduce:	1 Have a client in a game. 2 Perform a variety of inputs such as movement, shooting, throwing grenades and switching weapons.
Expected result:	The inputs should be performed in a reasonably quick time by the units (¡1s delay).
Actual result:	The inputs are performed in a reasonably quick time (¡1s delay).
Action:	None.

3.6 Death

Test ID:	3.6.1
Name:	Animation is played when a unit dies.
Description:	When a unit dies, an animation of the unit turning slowly white is played.
Steps to reproduce:	1 Have two clients in a game. 2 Get one client to kill the other.
Expected result:	The client whose unit dies shows the death animation.
Actual result:	The death animation successfully shows.
Action:	None.

Test ID:	3.6.2
Name:	Spectator system.
Description:	When a unit dies, they can spectate other units until they respawn.
Steps to reproduce:	1 Have two clients in a game. 2 Get one client to kill the other. 3 The dead client should click on the left and right arrow buttons that appear next to 'Spectating'.
Expected result:	The client whose unit dies should be able to cycle between units to spectate while they are respawning by clicking on the left and right buttons.
Actual result:	The client whose unit dies successfully can view other units via the spectator system. Cycling works well.
Action:	None.

Test ID:	3.6.3
Name:	Units respawn after a short time of being dead.
Description:	A unit respawns after a short time of being dead.
Steps to reproduce:	1 Have two clients in a game. 2 Get one client to kill the other.
Expected result:	The client whose unit dies should respawn after a few seconds of being dead.
Actual result:	The dead unit successfully respawns and full control can be taken of him.
Action:	None.

3.7 Sounds

Test ID:	3.7.1
Name:	Background music.
Description:	Background music plays while in game.
Steps to reproduce:	1 Have two clients in a game.
Expected result:	The background music is played on all clients.
Actual result:	The music is successfully played.
Action:	None.

Test ID:	3.7.2
Name:	Bullet sound when fired.
Description:	A sound is played when a bullet is fired.
Steps to reproduce:	1 Have two clients in a game. 2 Have both clients close together geographically in game. 3 Get one client to shoot. 4 Move one client far away and then shoot again.
Expected result:	The bullet sound should be played on both clients only when they are close together.
Actual result:	The sound is successfully played only when close together.
Action:	None.

Test ID:	3.7.3
Name:	Grenade explosion sound when grenade explodes.
Description:	A sound is played when a grenade explodes.
Steps to reproduce:	1 Have two clients in a game. 2 Have both clients close together geographically in game. 3 Get one client to throw a grenade. 4 Wait for the grenade to explode. 5 Move one client far away and throw another greande, wait for it to explode.
Expected result:	The grenade explosion sound should be played on both clients only when they are close together.
Actual result:	The sound is successfully played only when they are both close together.
Action:	None.

Test ID:	3.7.4
Name:	Reload sound when gun is reloaded.
Description:	A sound is played when a gun is reloaded.
Steps to reproduce:	1 Have two clients in a game. 2 Have both clients close together geographically in game. 3 Get one client to shoot and then reload their gun.
Expected result:	The reload sound should be heard on only the client who reloaded.
Actual result:	The sound is not played.
Action:	Investigate why the sound is not played.

Test ID:	3.7.5
Name:	Power-up collected sound.
Description:	A sound is played when a power-up is collected.
Steps to reproduce:	1 Have two clients in a game. 2 Have both clients close together geographically in game. 3 Get one client to pick up a power-up.
Expected result:	The pick up power-up sound should be heard on only the client who picked up the power-up.
Actual result:	The sound is not played.
Action:	Investigate why the sound is not played.

3.8 End Game

Test ID:	3.8.1
Name:	Assassins win on president death.
Description:	The assassins win when the president is successfully killed.
Steps to reproduce:	1 Have a client in a game with assassin AIs. 2 The user lets the assassin AIs kill the president.
Expected result:	A splash screen should appear informing of the assassins' victory.
Actual result:	A red splash screen is displayed telling of the assassins' victory.
Action:	None.

Test ID:	3.8.2
Name:	Escorts win on president safety.
Description:	The escorts win when the president is safely escorted to the safezone (the rainbow tiles on the map).
Steps to reproduce:	1 Have a client in a game no assassin AI. 2 The user guides the president to the safezone.
Expected result:	A splash screen should appear informing of the escorts' victory.
Actual result:	A red splash screen is displayed telling of the escorts' victory.
Action:	None.