Specifications [Version 3]

Alexandru Stoica - Daniel Robinson - Hongjia Xue
Shahil Pramodkumar - Ing Sam Yin - Ao Li -Tianxiang Song

Introduction

This document contains the specifications we have thoroughly gathered from our requirements. When we start developing codenames, we will use the specification to make sure the game is in line with the client's requirements. We have completed some diagrams such as a sequence diagram which was done by Alex and Daniel, a state diagram done by Shahil and Sam. Finally, some lo-fi prototypes planned by Anakin, Michael and Leo and done together.

Abbreviations and Definitions

Requirements and specifications can be traced as follows with their number:

- FR. Functional Requirement
- NFR. Non-Functional Requirement
- WS. Web Development Specification
- GS. Game Development Specification
- AIS. AI Development Specification
- *Italics* *Low Priority Requirement/Specification*

The following concepts are referred to in the specifications:

1. *Client* – the computer device directly used by the player.
2. *Server* - the computer that runs the game calculations and communicates with the *client.*
3. *Host* - the *client* that chooses the game options and is provides invite links to other players.

Specifications

The game can be split up into three sections for development, so that each area is encapsulated and can be worked on independently if needed. These areas are *web development*, *game mechanics* and *AI development.*

The specifications below, where possible, reference the requirement that they are fulfilling at the start of the sentence. I.e. [FR.19] covers Functional Requirement 19.

Web Development

1. [FR.19] The game should display a main menu when the game is opened.
2. [FR.13, FR.14] The main menu should either allow the player to select “Host Game” or “Join Game”.
3. [FR.14, FR.15] The host’s menu should provide an invite code.
4. [FR.14, FR.15] The invited player’s menu should accept an invite code.
5. [FR.18] The host should be able to select the following game options:
    a. ~~[FR.16, *FR,24*] How many players/AI players there will be.~~
    b. [FR.12] Whether there will be a bomb card.
    c. [NFR.21] The AI difficulty level (if applicable).
    d. [FR.9] Timer length.
6. [FR.18] Everybody (Including the host) should be able to select the following game options:

    a. [*NFR.5*] Accessibility options (I.e.: Font size and Colour).
    b. [*NFR.*6] Audio settings
7. [FR.17] Multiple spy players should be able to send & receive messages in a chat-box.
8. The client should store the game state.
9. [FR.2, FR.5] The client should be able send the game state to the server.
10. [FR.6, FR.7, FR.8, FR.10] The server should be able to calculate the new game state.
11. [FR.11, FR.12] The server should be able to send the new game state to the clients.
12. The game state should be shared in the specified format (see below).
13. [NFR.2] The client-server communication should be under `100ms`.
14. [NFR.1] The game should run on a web browser.
15. *[FR.22]* The game's CSS should have mobile compatibility.

Game Mechanics

1. [FR.1] The player should be able to play as either spy or spymaster by clicking on the corresponding button.
2. [FR.20] The game should accept 4 players.
3. [FR.2] Spymaster should be able to give a clue through a text box.
4. [FR.2] Spymaster should be able to select the number of related cards with a dropdown menu.
5. [FR.2] The client computer will check that the spymaster's clue is valid:
    a. [NFR.3] The spymaster should be able to type only one word in the text box.
    b. [NFR.4] The spymaster should not be able to type a word the same as card.
6. [FR.3, FR.3, FR.5, *NFR.6*] Only Spymasters should be able to see the coloured cards, hence the webpages will be different for both the spy and spymaster.
7. [FR.5, FR.8] Spies should be able to guess a word by clicking on the card they want.
    a. [FR.7] Spies can only select the number of cards + 1 that the spymaster has specified.
    b. [FR.8] Spies can end their turn early.
8. [FR.6, FR.7, FR.8, FR.10] The client keeps a score:
    a. The game should calculate the number of points based on cards chosen.
    b. The game should add the number of points corresponding to the number of cards picked from each team.
    c. Blank cards should not change the score but should end your team's turn.
9. [FR.11, FR.12] The game should end when conditions are met:
    a. When all cards from a team have been picked, the game should end.
    b. When the bomb card has been picked, the game should end.
10. [FR.9] All player's turns should be timed.
11. [*FR.23, NFR.6*] When the game is played locally, the game state can be saved.
12. [*FR.25*] The game should have a hint button for both spy and spymaster.
13. A player should be able to choose which team they would like to be on.

AI Development

1. [FR.1] The AI should be able to play as either spy or spymaster.
    a. The AI spy should get the predictions by comparing word vectors between hints and board words.

   b. The AI spymaster should provide hints by computing "scores" between possible hints (from vocabulary) and board words
2. [FR.1, FR.16, *FR.24*] One or two AI/AIs can be in a team.
   a. If two AI are in a team, they should take different roles and collaborate.
   b. If one AI is with a single player, the AI should assist the human player.
   c. Four AIs should play in two teams without human players, with one spy and one spymaster in each team. (Known as: *observer mode*)
3. [FR.21] The AI could be configured to achieve different difficulty levels.
   a. The AI difficulty should depend on the size of vocabulary or prediction accuracy.
4. [NFR.2] The NLP model of AI should be optimized to get the predictions under `500ms`.

Protocols

Client-To-Server

NOTE: THE NAME OF THE PROTOCOL MUST BE INCLUDED AS A HEADER

Name	joinGame
Headers	roomCode

Example	`GET / HTTP/1.1`
	`Protocol : "joinGame"`
	`RoomCode : "abcd1234"`

Name	joinTeam
Headers	Role Team

Example	`GET / HTTP/1.1`
	`Protocol : "joinTeam"`
	`Role : "Spymaster"`
	`Team : "Red"`

Name	createInitialBoardState
Headers	Timer Length Bomb Card AI Difficulty Number of AIs

Example	`POST / HTTP/1.1`
	`Protocol: "createInitialBoardState"`
	`TimerLength : "30"`
	`BombCard : "yes"`
	`AIdifficulty : "2"`
	`NumOfAIs : "3"`

Name	forwardClue
Headers	Clue Word Number of Guesses Player Turn
Example	`POST / HTTP/1.1` `Protocol : "forwardClue"`

	clue : “Tree” numberOfGuesses : “2” player : “red, spymaster” turn : “blue, spymaster”

Name	sendBoardState
Headers	Clue Word Number of Guesses Red Score Blue Score Timer Length Player Turn CardChosen
Payload	Cards
Example	POST / HTTP/1.1 Protocol : “sendBoardState” clue : “Forest” numberOfGuesses : “3” redScore : “3” blueScore : “1” timerLength : “30” player: “red,spy” Turn: “blue,spy” CardChosen : “3,4” “[red,potato,hidden][blue,dog,shown]; [neutral,apple,hidden][bomb,supercomputer,shown];” ...etc

Name	chat
Headers	message
Example	POST / HTTP/1.1 Protocol : “chat” message : “hi lol very pog”

Server-To-Client

NOTE: THE NAME OF THE PROTOCOL MUST BE INCLUDED AS A HEADER

Name	aiSpymove
Headers	Team aiDifficulty CardChosen Number of Guesses
Example	POST / HTTP/1.1 Protocol: “aiSpymove” AiDifficulty : “2” numberOfGuesses : “3” Team : ”red” CardChosen : “3,4” Cards : “[red,potato,hidden][blue,dog,shown];

	[neutral,apple,hidden][bomb,supercomputer,shown];” ...etc

Name	aiMastermove
Headers	Team Word hint Guess num
Example	POST / HTTP/1.1 Protocol: “aiMastermove” Team : ”red” Word hint : “fruit” Gusess num : “3”

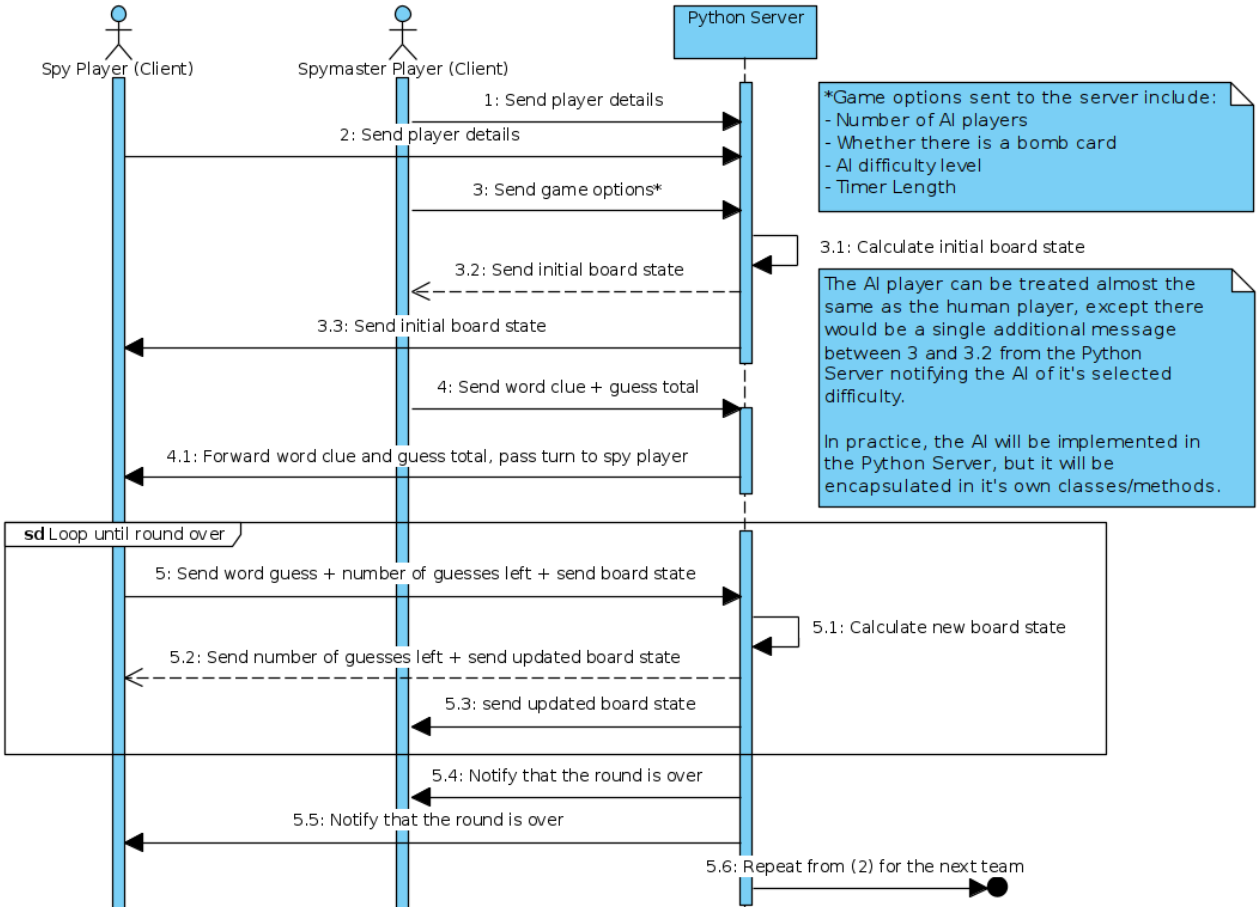
Name	receiveBoardState
Headers	Clue Word Number of Guesses Red Score Blue Score Timer Length Turn
Example	POST / HTTP/1.1 Protocol : “receiveBoardState” clue : “Forest” numberOfGuesses : “3” redScore : “3” blueScore : “1” timerLength : “30” turn : “red,spy” cards : “[red,potato,hidden][blue,dog,shown]; [neutral,apple,hidden][bomb,supercomputer,shown];” ...etc

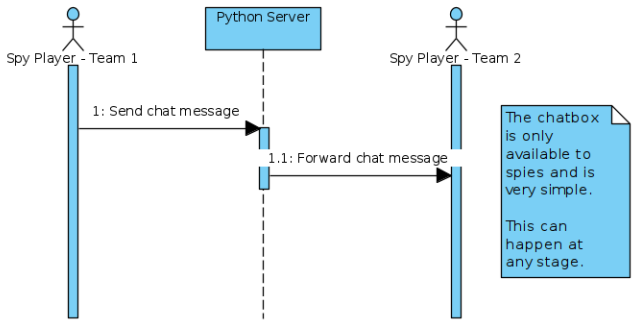
Name	isOver
Headers	Isbomb isCardAllPicked
Example	POST / HTTP/1.1 Protocol : “isOver” IsBomb : 0 IsCardAllPicked:0

…

Diagrams

Sequence Diagram





Note: For the purposes of part 1, the spymaster is assumed to be the host.

The first sequence diagram shows how the server will interact with the clients. Firstly, the game starts with the host selecting the options for the game (WS.5) and then the server will send the initial board state, which will be stored on the client's side (WS.8), to the players. After a move is made, the client sends the board state plus the changes made to the server (WS.9), the server will update the board state based on those changes (WS.10) and then it will send the new board state to all the clients (WS.11).

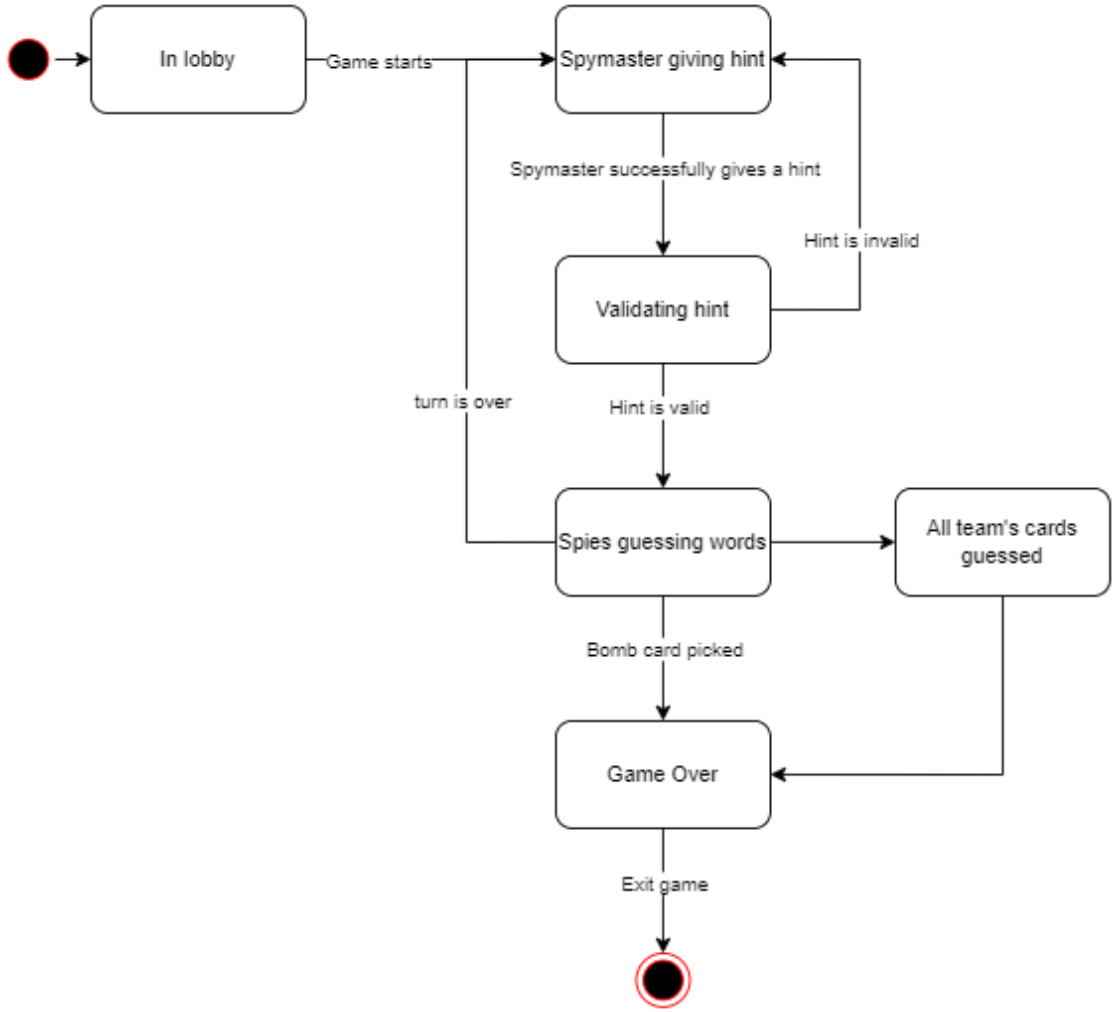
The functionality of the chat box is showed in the second Sequence Diagram (WS.7).

State Diagram

The state diagram shows the main states that the game can reasonably be expected to be in. This state diagram starts when either single player or multiplayer is selected from the main menu[WD1]. In the lobby, players can select how many AI players they want to play with or against and manage

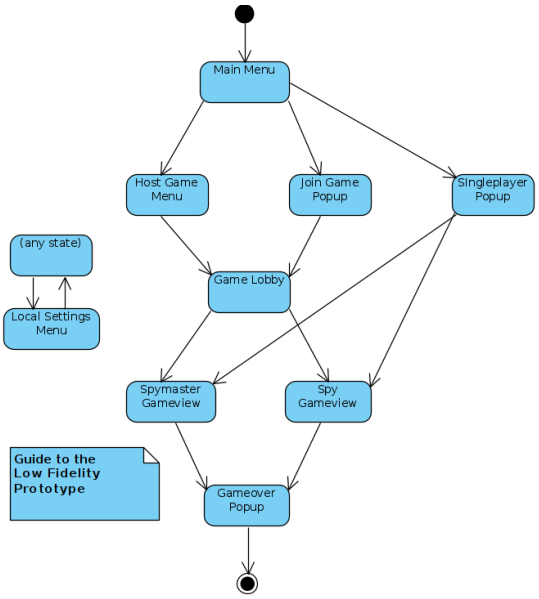
the rules for the current game[WD5]. When the game starts, there is a slight idling period while the spymaster for the first team evaluates their options. When the spymaster inputs a clue to be given to their spies, the game will validate it to check if the spymaster gave a legal word[GS5], being that it cannot be a word on the board. If the hint is valid, the spies will try to guess the word(s) hinted at[GS7]. Once the spy has chosen some card(s) or ends their turn early, the game will go back to an idle state as the other spymaster begins their turn.

The game carries on until either a bomb card is picked, in which case the game is instantly thrown into the game over state[GS9b], or until a team has no words remaining to pick, in which case they win[GS9a], and proceed to the game over state as well.



Low Fidelity Prototype

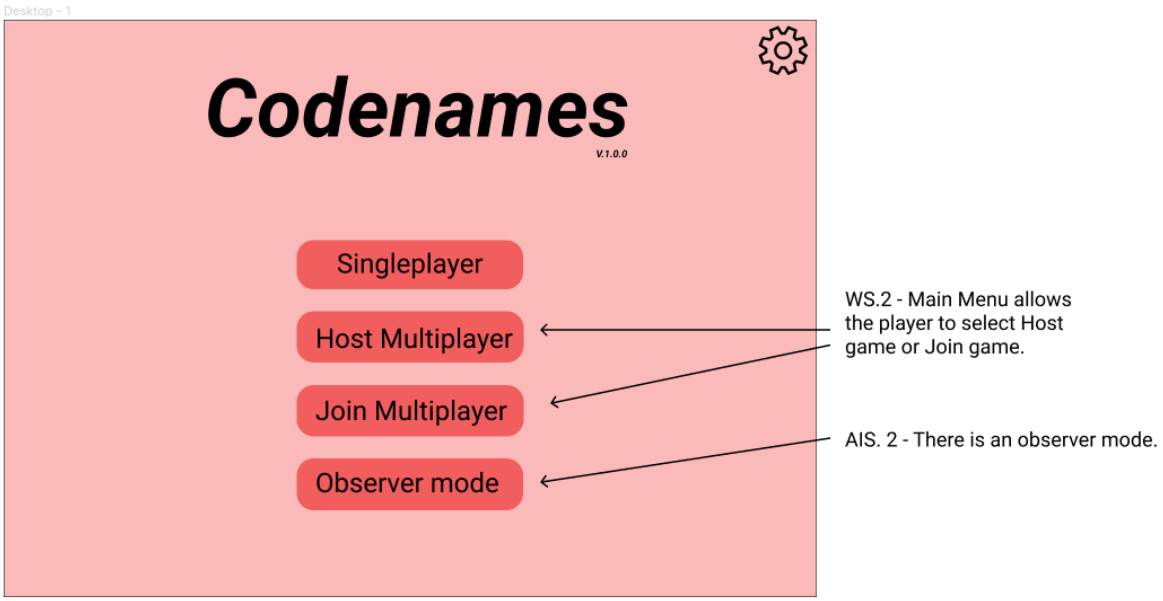
Navigation



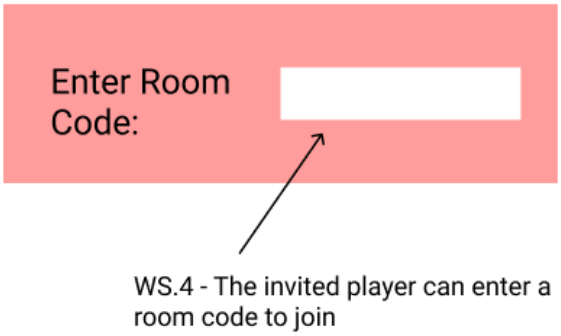
Main Menu

WS. 1 - Main menu is displayed when the game is opened.

WS.14 - The game runs on a web browser.



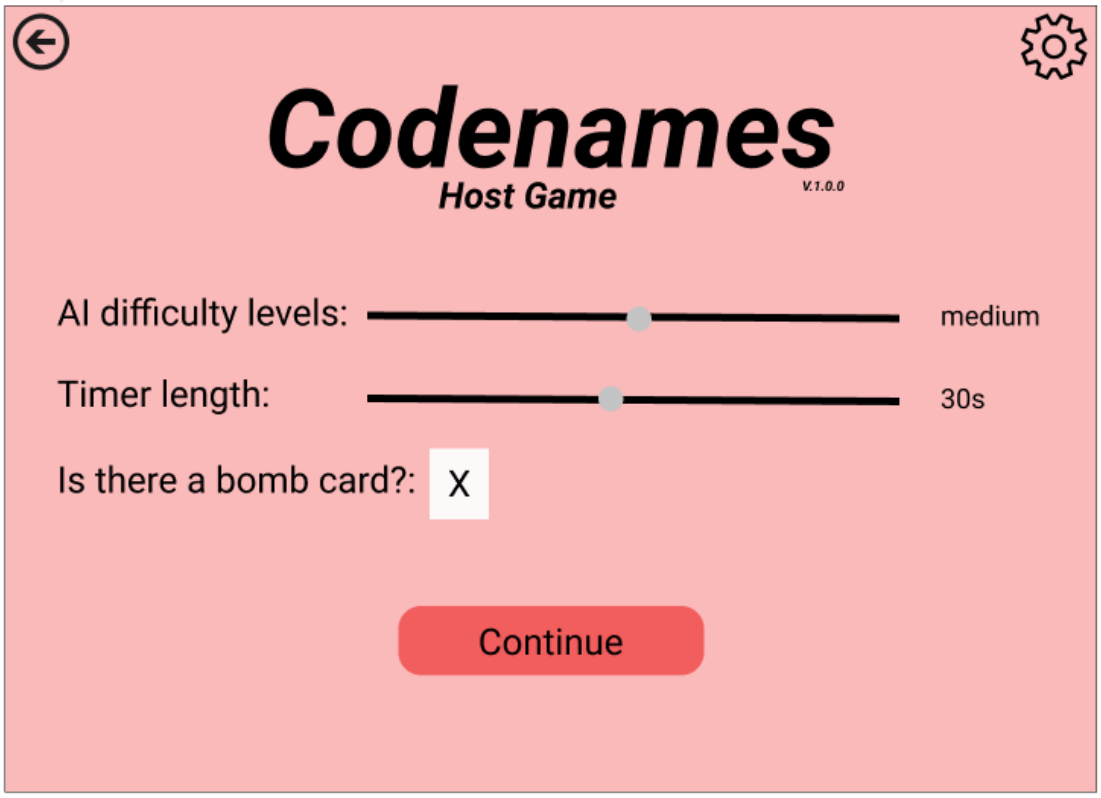
Join Room Popup



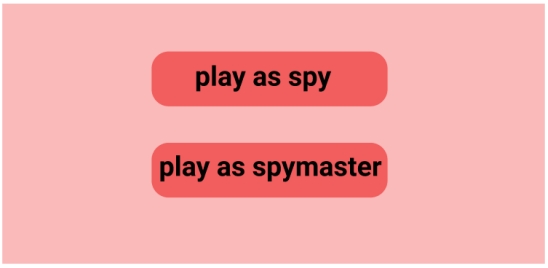
Host Game Menu

WS. 5 and AIS.3 - game options for AI difficulty, whether there is a bomb card and timer length

Desktop - 3



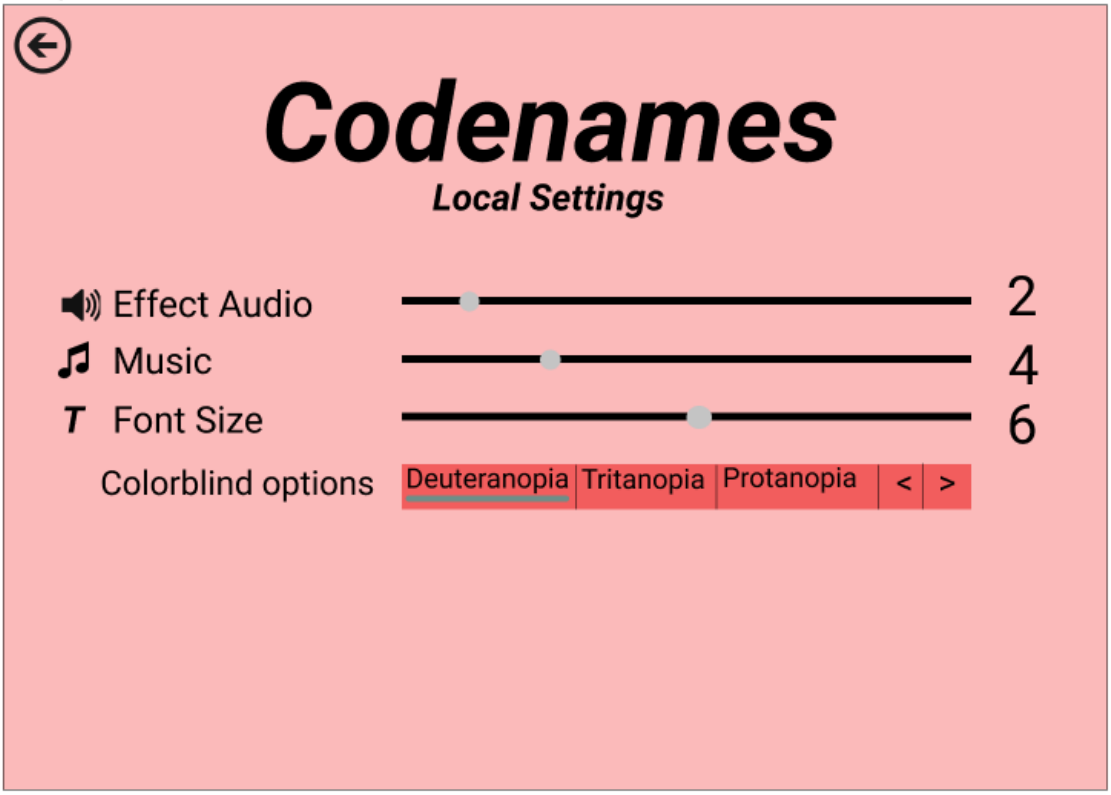
Single player Popup



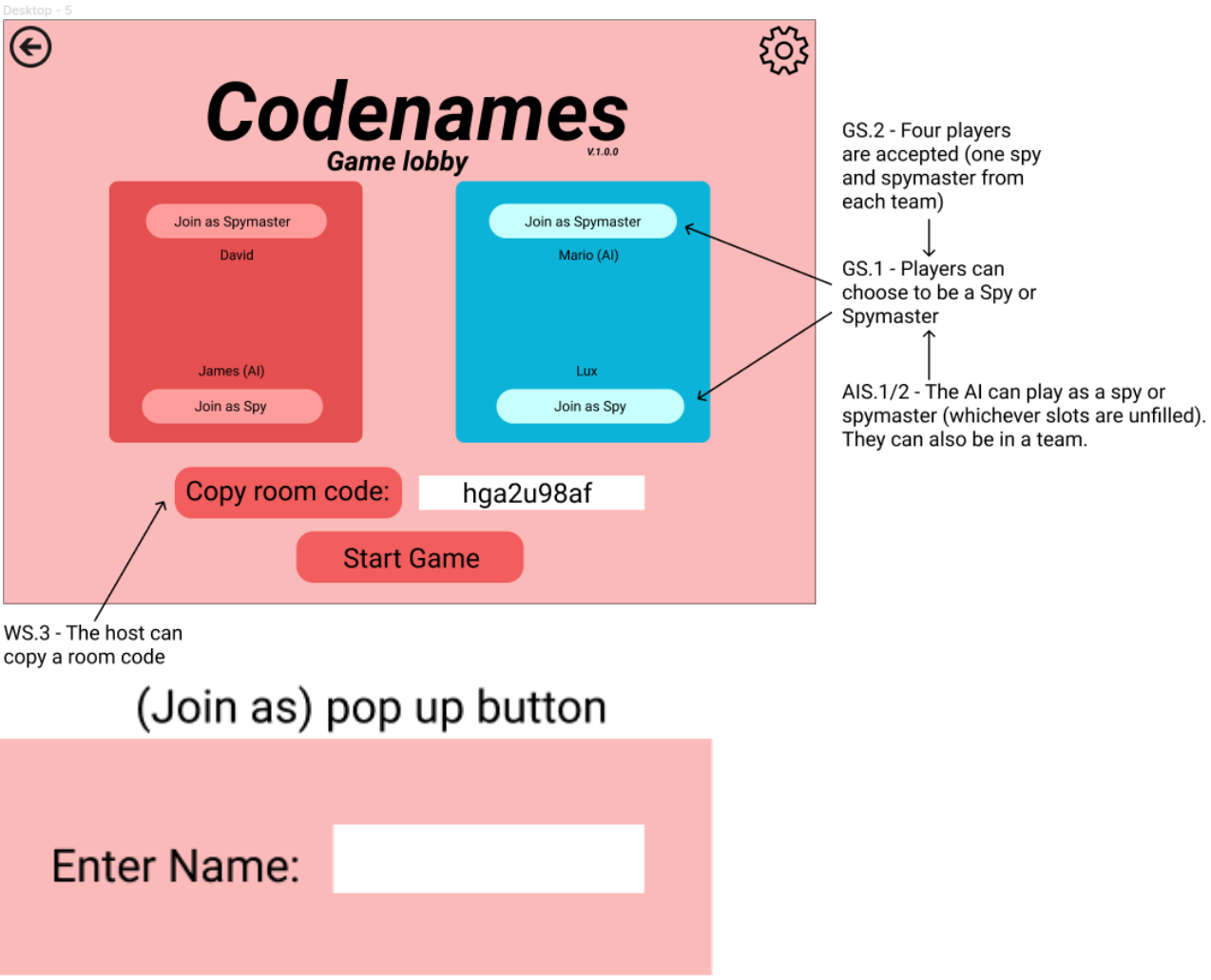
Local Settings Menu

WS.6 - Everybody should have accessibility and audio settings

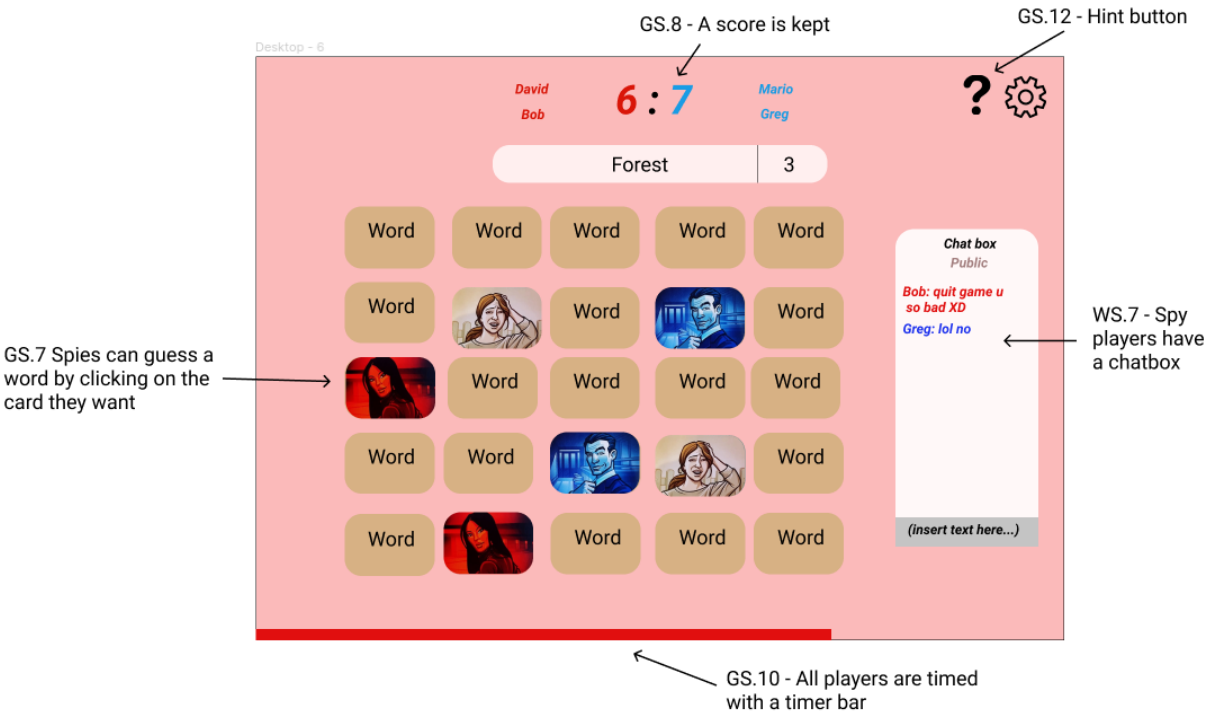
Desktop - 4



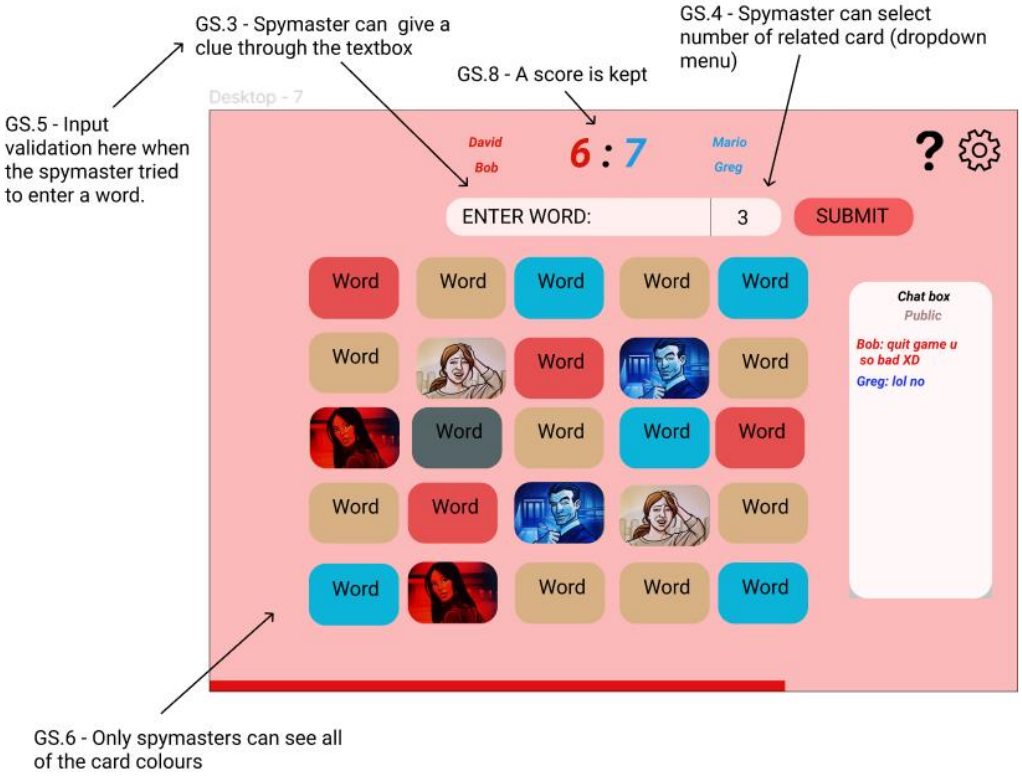
Game Lobby Menu



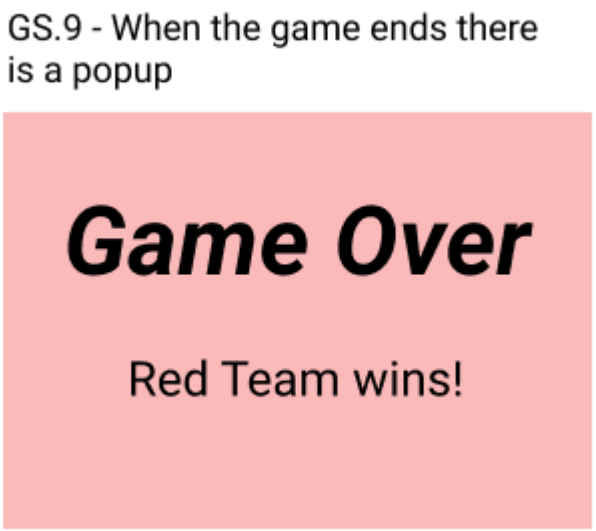
Spy Gameview



Spymaster Game View



Game-over pop-up



TO DO: change word backgrounds etc

1. Main menu
    a. Game title
    b. Button to tutorial
    c. Button to options menu
    d. Button to choose game mode
        i. Host multi-player

1. Waiting room screen
2. Choose role
3. Choose how many AI players (0-2)
4. Show code to invite others
5. Confirm button to show game screen for the chosen role

ii. Join multi-plater
1. Box to enter invitation code
2. Choose role
3. Confirm button to show game screen for the chosen role

iii. Single player
1. Waiting room screen
2. Choose role
3. Default 3 AI players
4. Confirm button to show game screen for the chosen role

e. Button to exit

2. Options menu
   a. Settings
      i. Sound
      ii. Text size
      iii. …
   b. Confirm button to go back
3. Game screen for spy
   a. Board cards
   b. Chat box
   c. Number of left cards for both teams
   d. Button to options menu
   e. Names and roles for members of both team
   f. Countdown in one tern
   g. Button to option menu
   h. Game log
   i. To surrender
      i. When all your teammates accept, show game over screen
      ii. If no human in your team, just show game over screen
4. Game screen for spymaster
   a. Board cards
   b. Box to type clue and number of words associated
   c. Number of left cards for both teams
   d. Button to options menu
   e. Names and roles for members of both team
   f. Countdown in one tern
   g. Button to option menu
   h. Game log
   i. To surrender
      i. When all your teammates accept, show game over screen
      ii. If no human in your team, just show game over screen
5. Game over screen
   a. Winner screen
   b. Loser screen

c. Button back to wating room

Specifications Validation

Internal Review

During an internal review the team decided to change the following to create the second version of this document:

- Removed WS.5a
- Added GS.13
- Changed state diagram
- Added note to sequence diagram
- Added single player prototype popup
- Changed game screen prototype

Further to the above, a list of Client-Server protocols were written to make version 3.

External Review

We reviewed this document with our supervisor and sponsor, Colin, who made remarks that the project seems well planned and everything has something to do. He said the technologies lined out in this document are sensible.

One of the areas that we needed to improve on was with the prototype which had limited mobile compatibility (*FR. 22).* The future design will include information in a sliding panel so that it can be slid out on mobile screens.