**Adam Fraser AH CS project**

**Multi-screen remote controlled asteroids**

**Project proposal**

**System overview**

this system is a modern take on the classic arcade game “asteroids”. While in the original users generally played alone only interacting with other users through a leader board this version will …

* Allow users to engage in multiplayer games across a large arena comprised of many screens
* Give users the ability to control their “ship” from any device with the latest browser technology but preferably a mobile / touch device
* Add lots of interesting game mechanics like powerups (e.g. speed boosts) and the ability to setup mining colonies on asteroids (give a certain number of points per cycle)

**End-users**

End users will likely be relatively experienced with computer systems due to the hassle of setting up the server and database. They will probably play the game in large groups since it requires many devices and will be most fun that way. They will need to have at least one device each to control their “ship” plus at least one additional device to run the server and act as an arena.

**Perceived need for system**

Currently there are no systems I am aware of that allow as many people as are available to play the classic arcade game asteroids across a large “arena” much less control it with their phones. this system will allow users to use whatever compatible devices they have to hand as a potentially giant arena to competitively play asteroids.

**Key features / objectives (scope?)**

* The game should be responsive with latency low enough over a local network to be unnoticeable
* There should be an intuitive way to initiate a game with other users
* There should be a simple to use and scalable leader board
* The game should display the score as it is being played
* The game should include interesting features like power ups, point boosts and obstacles
* The server should have a useful and intuitive dashboard showing such information as current number of games, total games and server uptime
* The server should keep extensive logs in an external persistent file
* The server should make use of a database to store scores
* The controller interface should be optimized for mobile devices
* The transition of entities from one screen to annother should be seamless or at least only involve a few ms of delay
* The asteroids automatically generated for the game should be “interesting” shapes and not just simple circles
* It should be easy for each player to follow their “ship” around the arena
* When a new powerup appears users should be alerted with a sound since it may be out of their sight
* Users should be able to attack one annother

**Restrictions**

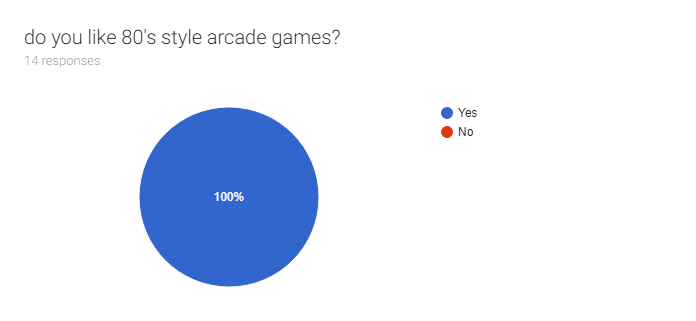
* The server needs to handle a socket for every controller and every display, if many players are sending acceleration requests at once the server may struggle to keep up. While this is not a physical limitation it means that the number of players who can play will be limited by the hardware the server runs on
* Since the entities on a display are processed by that displays device if many entities are on one display the device may struggle to keep up
* The controller is designed for a touch interface. On devices like hybrid laptops where the touch refresh rate is lower and the touch capacitor’s more spaced out the experience may be jittery to say the least

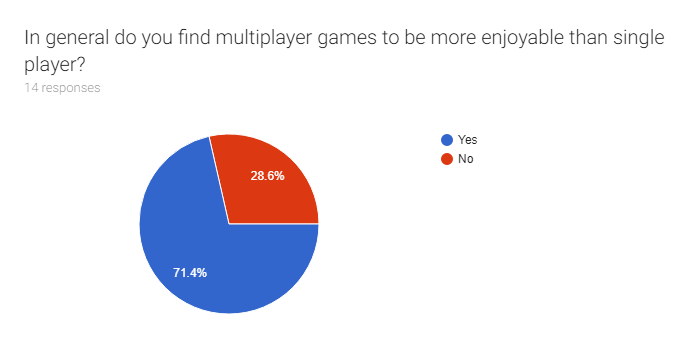
**Meeting requirements**

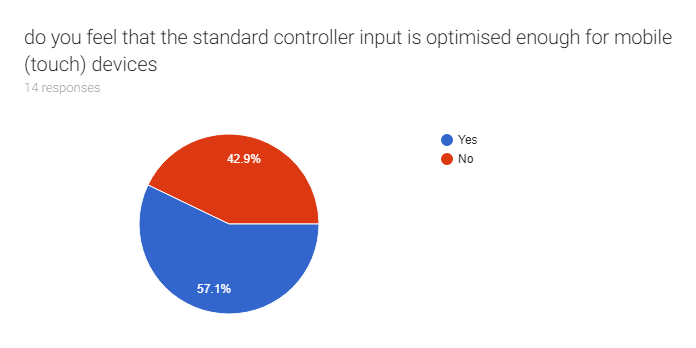
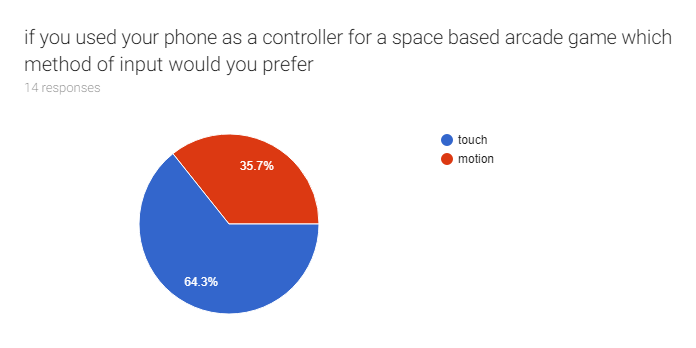
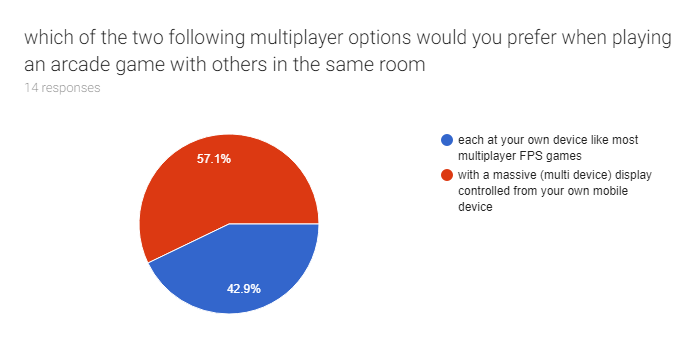
* This program will write to an external log file
* this program will process data from a mongodb database in a simmilar way as a SQL based database may be used
* this program will use 2D arrays to efficiently check for collision
* this program will use HTML and javascript for the front end
* it will also process form data submitted from clients
* it will make extensive use of modularity
* this program will use a linked list to store sockets of arenas

**Initial research**

**User surveys (MISSING BAR CHART)**







are there any features you think are especially important for a multiplayer game that have not been mentioned (not required)

|  |
| --- |
| The ability to taunt other players |
| Some way to mock the loser in the game to save me the effort of doing it in real life |
| It would be nice to join a game part way through |
| Playing with other people |

**Feasibility study**

**Criteria for cancelling the project**

the project should be cancelled or rethought if any of the bellow criteria are met

* deadlines are consistently missed
* a serious technological barrier is met e.g. the hardware required to run the displays with low latency is not consumer grade

**alternative solutions**

while there are many other implementations of asteroids some with multiplayer functionality I am not aware of any implementation that works quite like this one by allowing players to use their phones as controllers and as many devices as they want like a massive arena.

**Financial feasibility**

The system will cost nothing but time and will not be sold. In order to adequately test the system several mid-tier consumer grade PC’s will be required to act as the arenas and several mobile devices will be required this could be very expensive but there are computers available in school and there are enough spare mobile devices to hand.

**Legal feasibility**

There may be concerns in relation to the data protection act and GDPR due to the leader board which uses a database. If users enter their real name in the leader board the information may be personally identifiable

Users will need to be made aware of their responsibility to conform with the communications act when they are entering their leader board name so they know not to enter obscene names

**Technical feasibility**

due to the nature of the game it will be vital for there to be a low latency between controller, server and arena it should be so low as to be unnoticeable and ideally it should be around the time it takes to compute a single frame from update to render. If it is not possible to keep latency low enough a serious re-evaluation will need to take place

the most powerfull hardware I have easy access to is a 7th gen core i5 laptop with 8GB of RAM this should be enough to run several arenas and many controllers (serving the same number of users) but there may be serious performance and even stability issues with larger numbers. If the available hardware is not enough to run the server even for small numbers then one of 4 things will need to happen (in order)

1. try to make my code more efficient (this will probably not be hard since my code usually comes with tons of baggage)
2. try a different architecture
   1. move more of the burden from the server to arenas and controllers
   2. cloud based, this would probably have serious latency issues
   3. decentralised computing with multiple servers, this would be an absolute nightmare to implement and I may not have the neccesary skills to do it
   4. distributed computing with no server and more of a mesh structure, I definitely don’t have the skills to do this but the neccesary skills may be attainable in the allotted time
3. re-evaluate the project however if this is not feasible due to deadlines
4. look into getting more powerfull hardware, this is a last case resort as it would be prohibitively expensive to buy brand new hardware

since the burden of physics and graphics is entirely on the arena’s there may be difficulty in ensuring they are able to handle multiple users at once especially if the hardware of the arena’s are vastly different.

**Constraints**

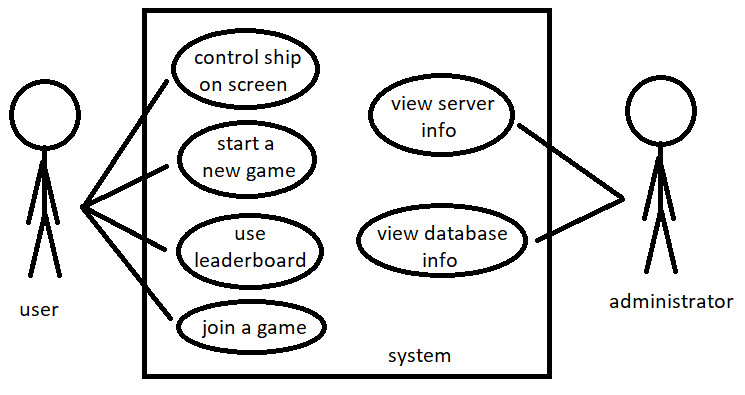
The only time money should be spent directly on the project is in the final research (evaluation) stage to run a server capable of hosting many players and arenas this may not be neccesary, but should it become neccesary the cost should not exceed a total of £50

The project should at no point be more than a month behind, if it is more than a month behind and it is unlikely that the project will be brought back in line with the project schedule a serious re-evaluation will need to take place possibly leading to the removal of some features

The project should be modular and well commented enough to be readable by anyone with a reasonable level of experience with node.js and it’s common libraries. The pseudocode should be readable by anyone with a reasonable level of experience in any language, if this is not the case then work will need to be done to ensure that the code is readable and not just garbled spaghetti mashed out on a keyboard.

**Functional + operational requirements**

**System use case analysis**



**Requirements specification**

**Design**