011-01
Mr. Krabs, he was number 1!
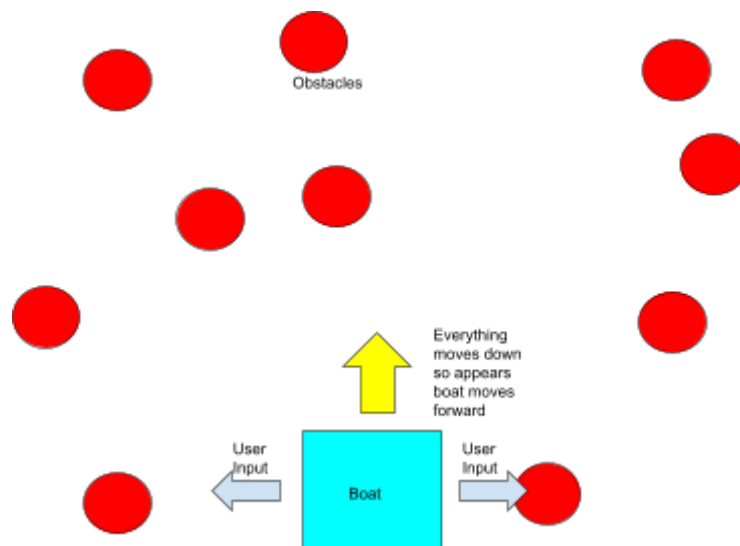Kylie Elbert
Jules Geneser
Riley O'Byrne
Chelsea Stockberger
Davis Cohen
Duke Manchester

Application Name:

-   DON´T CRASH

Application description:
-   Boat or car is trying not to crash into obstacles that keep coming from one direction continually going faster and faster. The player will have a birds eye view of the game as if they are looking down at the vehicle. The game score will continue to go up until the Boat or car crashes, at which the game ends and its score leaderboard appears.

Vision Statement:
-   "To make a simple game a daily game for everyone of all ages"
-   For anyone
-   Who likes to play simple games when they're bored
-   The game is an easy to play distraction.
-   That gives players' something easy to focus on when they need to take their mind off stuff.
-   Unlike a complicated, high stress game like Call of Duty,
-   Our product is fun for everyone.
-

Version Control:
- https://github.com/CU-CSCI-3308-Fall-2021/CSCI-3308-Fall21-011-01.git

Development Method:

Jira:
https://csci-3308-fall21-group1.atlassian.net/jira/software/projects/G00/boards/1/roadmap?atlOrigin=eyJp
IjoiMjFlODI3MzgzOTljNDU2MGFmOTU4ODhmOTIyZGNlNDEiLCJwIjoiaiJ9&selectedIssue=G00-7

Communication Plan:
- For this project, everyone in the group is in a group text message. We meet weekly in person in the Engineering Building. When we work on documents together we use google drive and share one document to everyone in the group
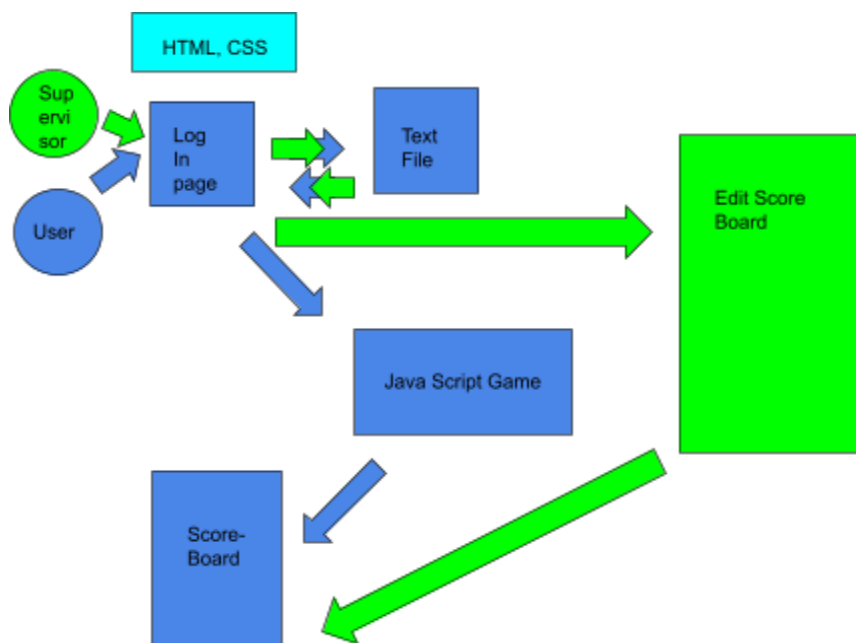
Meeting Plan:
Mondays 3:30 - TA Meeting
Mondays 5 - 7ish: Team meeting in Engineering Center

Proposed Architecture Plan:
Link to JIRA Board: https://csci-3308-fall21-group1.atlassian.net/jira/software/projects/G00/boards/1
- HTML for the web page (information and glue that holds things together)
- CSS to stylize the page
- JavaScript for the actual functionality of the game.
- Text file to store usernames and checks log in status

Use Case Diagram:

uc Use Cases

System Boundary

Waiter — receive order — Order Food <<extend>> Order Wine

place order

confirm order

Serve Food

Cook Food — Chef

<<extend>> {if wine was ordered}

Serve Wine

Client — Eat Food <<extend>> Drink Wine {if wine was served}

facilitate payment

pay

Cashier — accept payment — Pay for Food <<extend>> Pay for Wine {if wine was consumed}