Team Roles:

- Chairperson : Vanshika Verma - 21327286
- Scribe : Maheen Ahmed - 22305226
- Timekeeper : Bokang Ramatla - 22359691

High Level Design - Research & Decisions:

Our team divided research questions between us to answer individually. We then presented our findings to one another during our first meeting.

Here is our initial collective research of high level design & how it relates to our project:

Components & Interaction

Organisational system is basically the structure of the overall game; this includes how all the components will interact with each other and their individual structure.

Components : Board, Logic, UI, Saving and Loading, Player

Interaction:

- **Hexagonal Board and Game Logic :**
  - Structure of the game stage - hexagonal board + positions.
  - logic will be processing the player actions
  - when player makes moves the logic needs to calculate the outcome based on the rules &
  - update the board by managing the state of rays and atoms.

    May look like :

  - Ray Placement: experimenter places ray on board
  - Ray interaction : game logic calculates based on atoms position(placed by setter beforehand) if ray is directly hit, absorbed, deflected or reflected.
  - Game logic updates : board changes to visually represent experimenters move (marks where the direct hit, absorbed, deflected or reflected occurred).
- **UI and Player (User interactions):**
  - UI takes in inputs and displays results.
  - The way players will interact with the ui is to make a move, receive feedback as a result of the move made.
- **Saving and Loading and Game Logic** :
  - Interact with game logic to store and get game state.
  - ensures each round is saved & the scores are getting saved.
- **Game Logic and Player :**
  - Logic processes players decision and updates
  - provide rules of the game to the player

- provide the outcome of players action to the ui for display

◆ These are all the components we must consider in our software architecture.

What are the software architectures best suited for this project?

We aim to use an MVC architecture as the main architecture for this project.

We feel MVC architecture is best suited for this project due to these points:

- Has good maintainability & testability (good layer abstraction & decoupling).
- Facilitates the use of efficient code (have reusable code where possible - util sub-packages).
- easy to understand and well structured.

We feel that Hexagonal (Ports and Adapter) architecture may be better suited to a larger, more complex project where it might be beneficial to have several hexagons (domains). For example, if this game required network connectivity between players, or it was to be hosted online/on several different platforms. We feel implementing Hexagonal architecture for this project may lead to some over-complication, whereas MVC architecture provides sufficient layer abstraction for a project of this scale. Between Onion architecture and MVC; we chose MVC, again, for simplicity and suitability.

TUI/GUI?

We have decided to use a GUI for this project due to the graphics requirements and feel it would make for better UX.

GUI best suited for this project?

We have decided on JavaFX as our GUI for this project. We felt libGDX would be better suited to a game with more complex graphic requirements. JavaFX fulfils our requirement for 2-d graphics & is a more modern choice than Swing.

JavaFX also is in line with the non-functional requirements of our project;

- portability/installability (cross platform & jpackage/jlink for standalone package)
- maintainability (OpenJFX community support + releases alongside JDK)

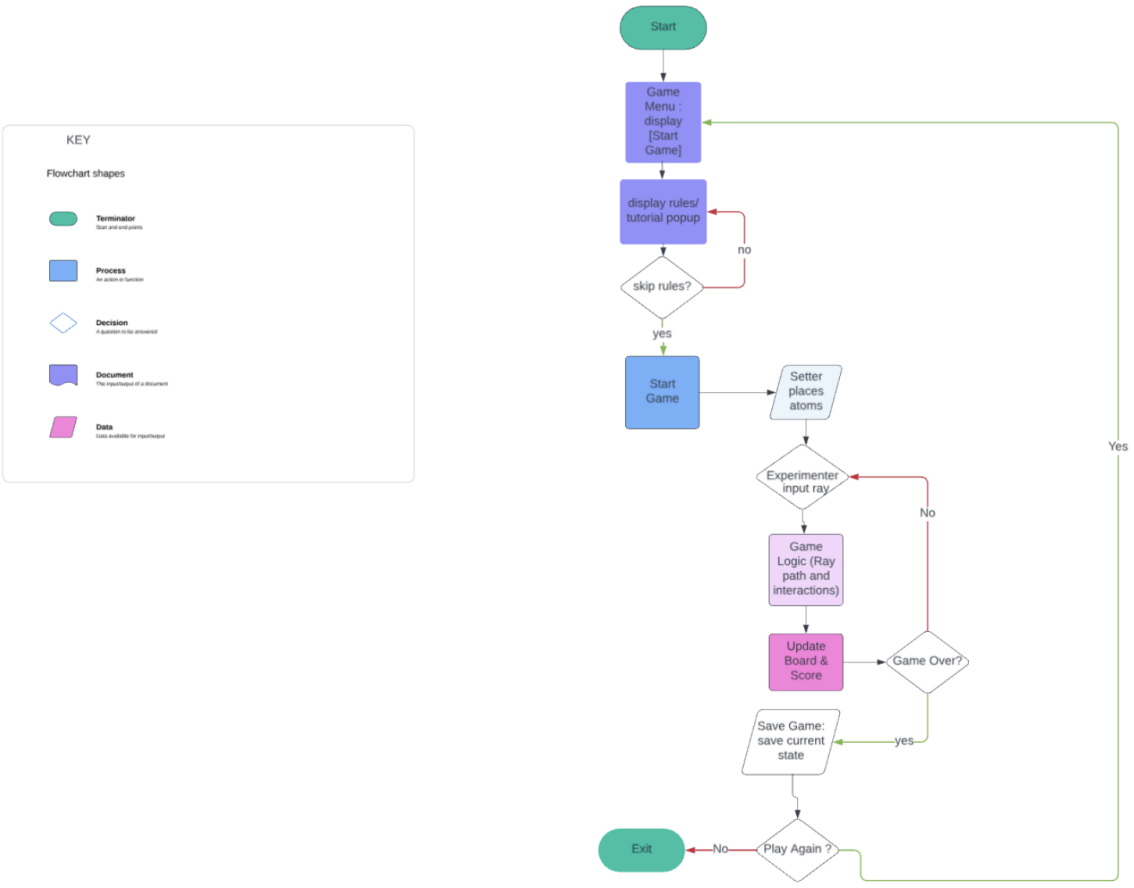
High Level Design - Diagrams:

High Level Design tools used: diagrams.net/draw.io, LucidChart, Trello.

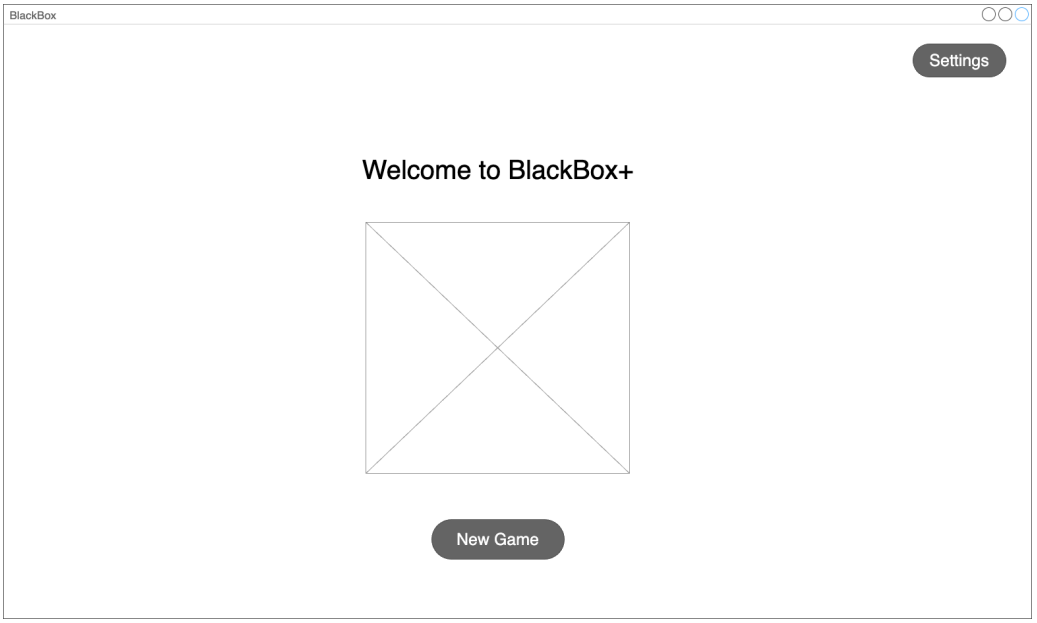
We decided on draw.io & LucidChart for the reasons outlined below:

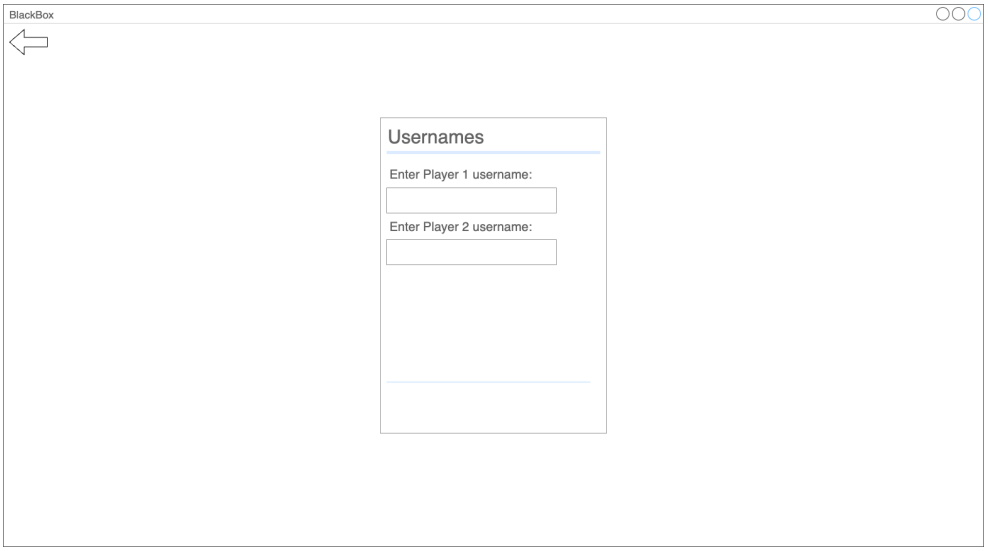
- Web based
- Collaboration feature
- Sufficient features for our needs:
  -> UML format design
  (wireframes, flow charts, use case diagrams, class/object diagrams).

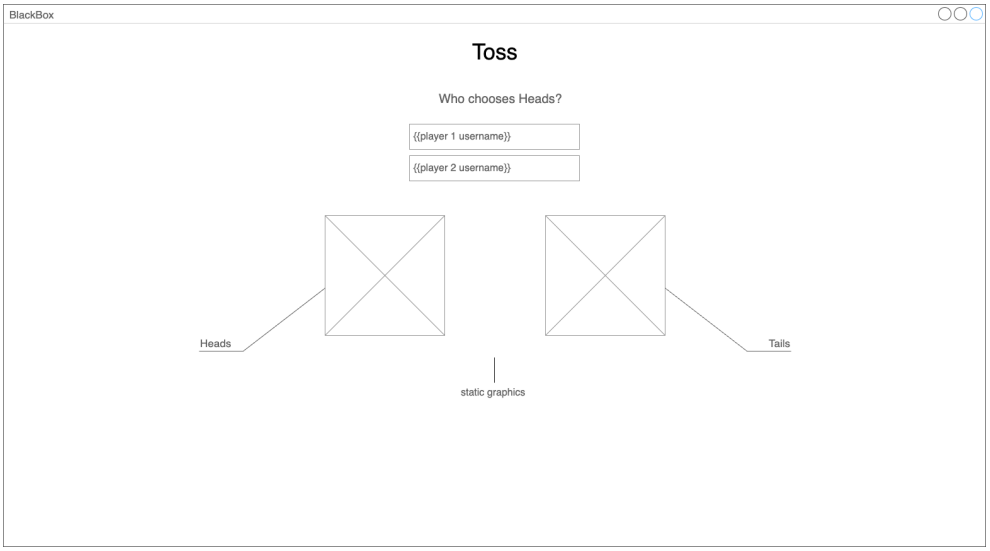
- **Flowchart (Model) - helps us to see workflow & sequential steps in the processes of our system**

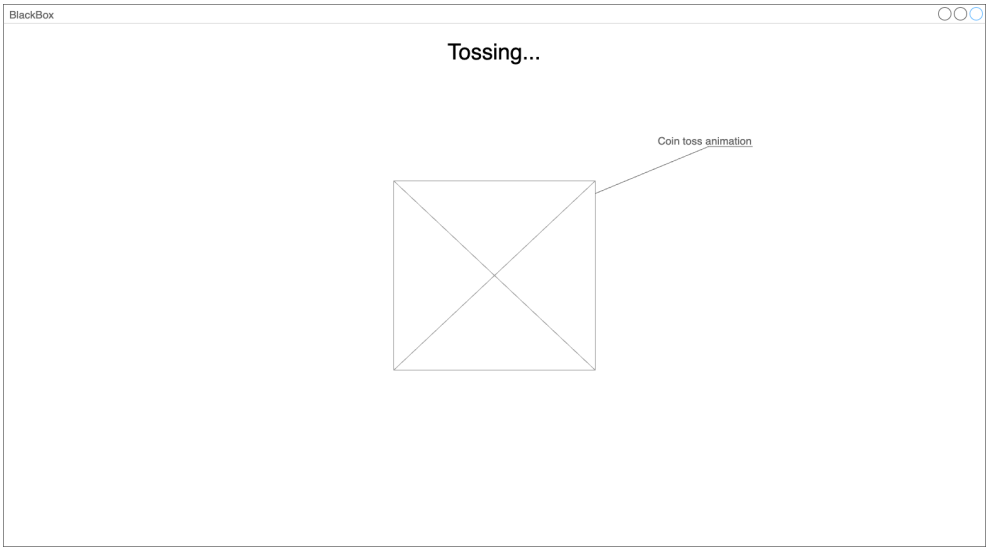


- **Wireframes - helps us visualise the UI (view) as well as realise any missing features.**




BlackBox
Usernames
Enter Player 1 username:
Enter Player 2 username:


BlackBox
Toss
Who chooses Heads?
{{player 1 username}}
{{player 2 username}}
Heads
Tails
static graphics


BlackBox
Tossing...
Coin toss animation

BlackBox

{{player username}}
WON THE TOSS

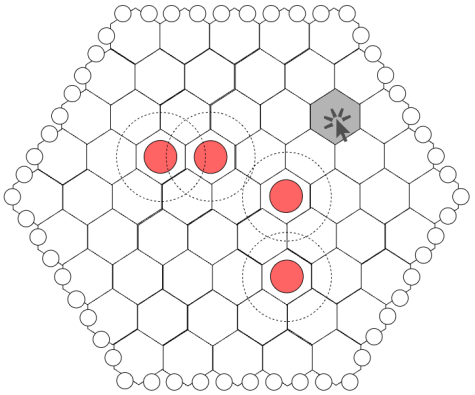
Choose:

Setter | Experimenter

BlackBox

Rules

Setter: {{username}}

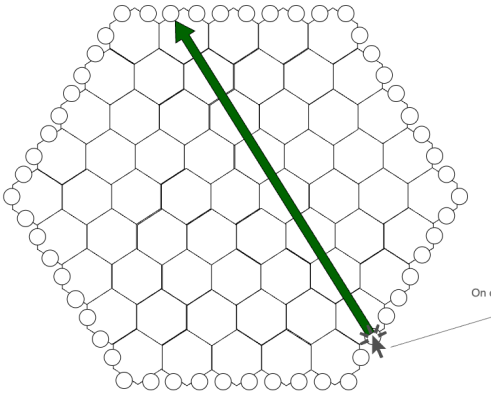


Ready

BlackBox

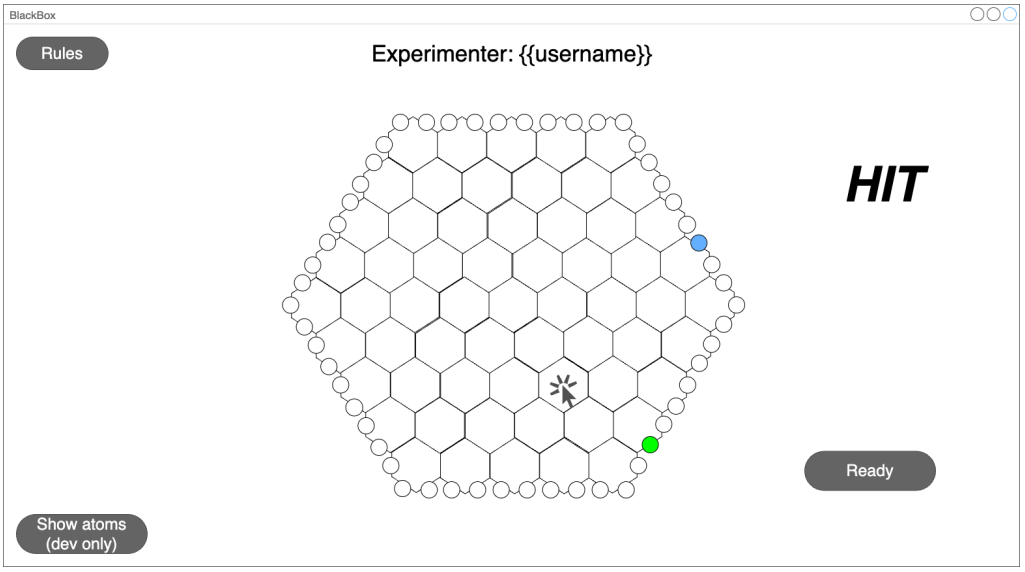
Rules

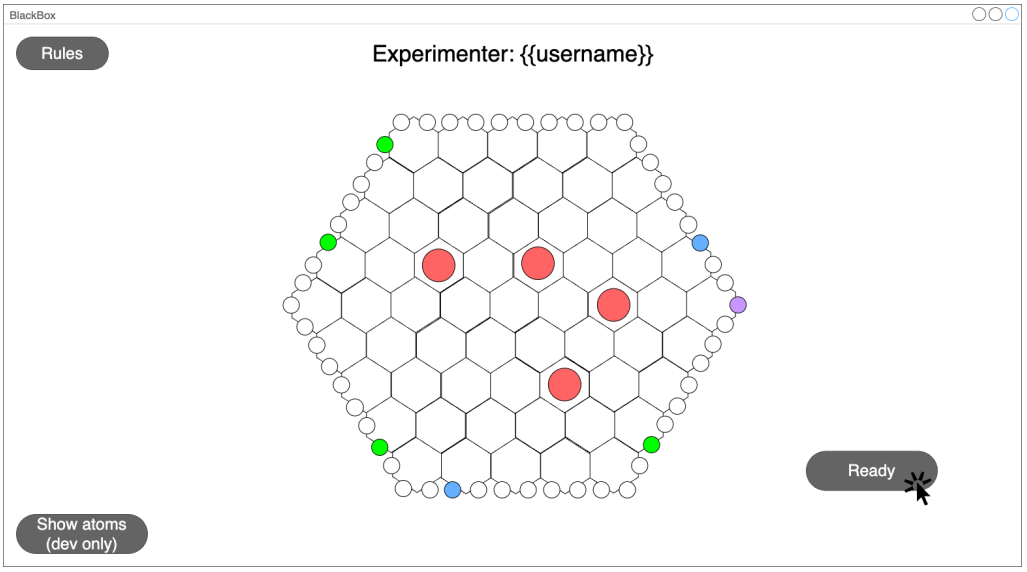
Experimenter: {{username}}



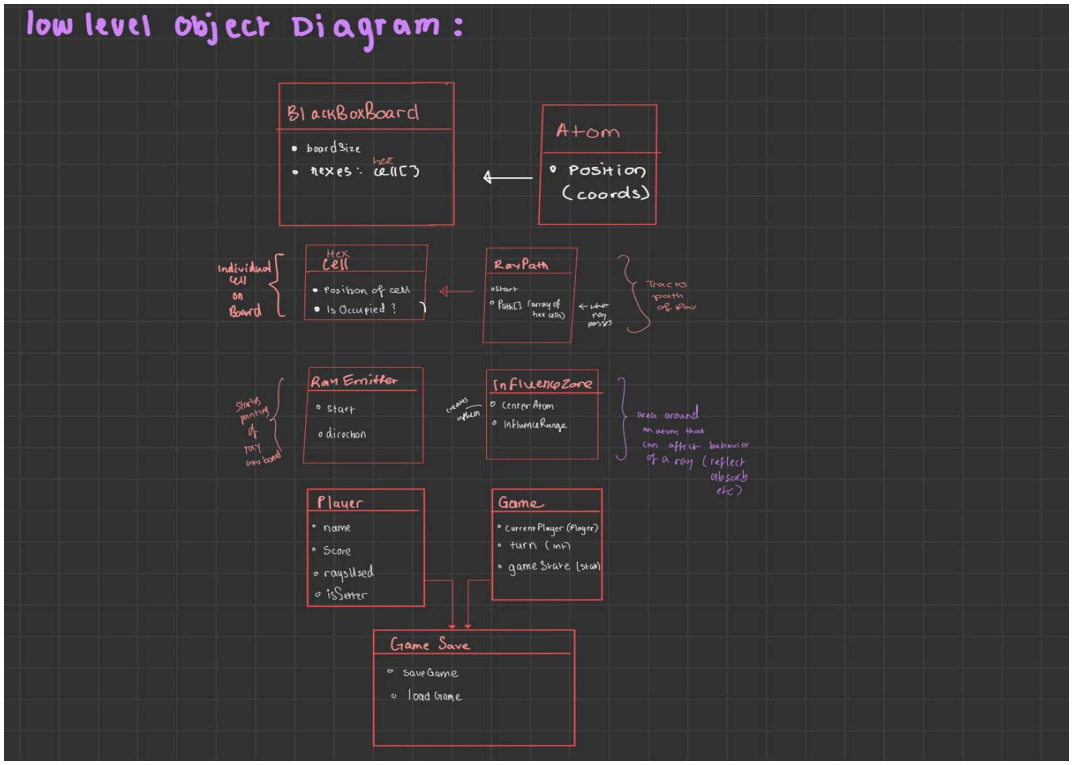
Fire Ray

Show atoms
(dev only)





- **Object diagram - helps with identify classes/objects in the core game logic (Model).**



View/UI flowchart: will help in identifying View and Controller classes.