Sprints:

Sprints	Features
1	**1.Set up** JavaFx project/library & structure with M-V-C architecture. **2. Model** - GameBoard Class: ● Complete BlackBoxBoard class (implements main game stage hexagon). ● Complete Hex class for individual hex cells. - Atom Placement: ● Implement Atom class. ● Add method in Gameboard to place atoms at random positions. **3. Controller** - Implement controller classes that handle the inputs and outputs of the model classes above. **4. View** - Implement an extremely basic text representation of the output of the model classes in the console, printing out the result for testing purposes. **5.Visibility:** ● Develop functionality to show or hide atoms and the area of influences ● Unit tests
2	**1. Model** - Ray class ● Implement Ray class to calculate and store - entry point and path. - Ray Absorption ● Add to Ray class direct hits and mark them as absorbed. **2. Controller** - Implement controller classes that handle the inputs and outputs of the model classes above. - Implement controller classes to handle View classes. **3. View** Begin a basic implementation of the main gameboard UI in JavaFX (GameBoardView class). ● Include grid & atoms. ● Unit testing in each layer.
3	**1. Model** - Ray Deflection : ● Update Ray class to deal with 60 degree deflections near atoms. ● 120 degree deflection as well when the ray passes two atoms.

	- Ray Reflection: ● Logic for ray reflecting off atoms located at the edge of the board. - Complex Path ● Calculate other complex ray paths. - GameState class ● Implement GameState to keep track of current state of game (scores, no. of rays used, game status). **2. Controller** - Implement controller classes that handle the inputs and outputs of the Ray class & GameState class. - Implement controller classes to handle the View classes. **3. View** Continue the basic implementation of the main gameboard UI in JavaFX (GameBoardView class). ● Should now include rays in the grid. ● Unit testing in each layer.
4	**1. Model** - Functionality for experimenters to announce their guesses and reveal atom locations. - Implement Player class. - Implement classes for the start screen, game over screen and any dialog boxes. - Implement additional features such as usernames. **2. Controller** - Implement controller classes that handle any new interactions. - Implement controller classes to handle the View classes. **3. View** Completely implement the View layer. ● Implement additional features according to wireframe (coin toss, customisations etc). **4. Documentation** - Ensure correct and structured documentation. ● Unit testing in each layer.