Date: February 20, 2025

First Project Meeting and Idea Discussion

What did we do today?
Today, we had our first project meeting. We spent time chatting and getting to know each other's strengths, which was important for task distribution.

Meeting Summary:
- Everyone shared different project ideas.
- We discussed the examples provided in class and analyzed them together.
- By the end of the meeting, we reached a common decision on how our project would be structured.
- Each team member was assigned the task of conducting general research on the project.

Future Plans:
- We scheduled our next meeting for **Tuesday, February 25, at 10:00.**
- In this meeting, we will share the results of our research and distribute specific tasks accordingly.

Date: February 25, 2025

Project Meeting and Task Distribution

What did we do Today?
Today, we held a meeting with our project team on Discord. The main purpose of the meeting was to share the research we had done, clarify task distribution, and discuss how to best prepare for the first milestone.

Meeting Summary:
- Everyone presented their research on the project, and we evaluated the gathered information.
- We assigned specific tasks to each team member.
- We created a plan outlining the steps needed to complete the first milestone.

Challenges We Faced:
- A change needed to be made regarding the game characters. Everyone had different ideas, but by the end of the meeting, we reached a common agreement on how the characters should be designed.

Future Plans:
- Everyone will focus on completing their tasks until the next meeting.
- Our next meeting is scheduled for **Thursday, February 27 at 12:00**. In the meeting, we will share our progress with each other.

Date: February 27, 2025

Project Meeting and preparation for the presentation

What did we do today?
Today, we discussed our Game name and decided to name our game: **Think Outside the room**.
We talked about our completed game concept and worked on our presentation for the next week.

Meeting summary:
- We decided our game name: **Think Outside The Room**
- We assign who is responsible for what in the presentation.

Future Plans:
Everyone will work on the presentation

Date: March 7, 2025

Project Meeting and preparation for the presentation

What did we do today?
Today, we held a meeting to discuss Milestone 2 and distribute tasks among the project team for it.

Meeting summary:
- We talked about Milestone 2 and how to prepare for it more effectively.
- Task distribution was finalized for the team, ensuring everyone has a clear role.
- We discussed strategies to improve our workflow and meet the milestone requirements efficiently

Future Plans:
- Team members will begin working on their assigned tasks.
- .A follow-up meeting will be scheduled to discuss the challenges if necessary.

Date: March 17, 2025

Chat Feature Testing and Command Standardization

What did we do today?
Today, we tested new functionalities for our game and improved our project by working collaboratively, ensuring smooth integration and quality time.

Meeting summary:
- Aisha coded the **chat feature**, and we tested it together to ensure smooth functionality.

- Illia created a standard for sending commands to the server, defining how the server processes and handles these commands. He explained the implementation with examples.
- Using this new standard, we successfully integrated the **nickname changing** feature, allowing players to change their nicknames through the server.

Future Plans:
- Continue refining the chat system and ensure commands are properly handled by the server.
- Plan another session to test and validate more **server-side** functionalities.

Date: March 20, 2025

Project Meeting and preparation for the presentation

What did we do today?
We merged the **login/logout branches** and improved the login functionality in our game.

Meeting summary:
- The **login system** was implemented to create a player and enable keybinds without assigning a random player.
- The following **server commands** have been tested and are now functional:
  create -> Creates player
  login -> Assigns an ID to the created player object
  logout -> Deletes the player from the session.
  exit -> Terminates the client.

Future Plans:
- Plan another session to review and test additional command functionalities.

Date: April 2, 2025

Task Distribution for Milestone 3

What did we do today?
Today, we held a team meeting to plan for Milestone 3. During the meeting, we discussed the main components of the milestone and assigned responsibilities across the team.

Meeting summary:
- We went over the different aspects of Milestone 3, including development, testing, documentation, and presentation.
- Tasks were distributed based on individual strengths and previous contributions

- We made sure that every team member is involved in at least one area, and most responsibilities are shared to encourage collaboration.
- Special attention was given to key areas such as game logic, GUI, protocol, login and lounge systems, as well as the final presentation.

Future Plans:
- Everyone will begin working on their assigned parts of Milestone 3.
- Open questions (like a few undefined features) will be clarified in upcoming meetings.
- We'll continue to collaborate closely and track progress to stay on schedule.

Date: April 3, 2025
Author: William

Argument Parsing to the Main class

What did we do today?
- Today I implemented a command-line argument parser so that users can specify the server/client mode, IP, and port when launching the game.
- I introduced a Rope mechanic into the main class with reverse kinematics, laying groundwork for the item rope.

What did I learn?
- how to do reverse kinematics and how to work with argument parser.

Challenges
- Ensuring the argument parsing did not conflict with JavaFX startup.
- Integrating a Rope concept without fully defined physicsä
- Making sure that the JavaFX environment initializes in the correct order, without Exceptions.

Future Plans
- Synchronize the rope

Date: April 4, 2025
Author: Sena

GUI Implementation for Chat and Lobby Features

What did we do today?
Today Illia and I worked together on implementing several key features into the game's graphical user interface (GUI). These features are essential for enhancing the multiplayer experience and communication within the game.

Meeting summary:
- We designed and integrated a multi-lobby system, allowing each game to have its own dedicated lobby along with a separate internal chat.

- We implemented a way to list players within each lobby, so users can see who they are playing with.
- Additionally, we developed the Whisper, Global, and Lobby Chat tabs as part of the chat system UI.
- While the features are not fully functional yet in terms of backend communication, we successfully built a clear, user-friendly interface that is ready to be connected with the underlying server logic.

Challenges We Faced:
- Since the server-side logic for these features was still under development, we had to simulate behavior and make assumptions about how data would flow between the client and server.
- Ensuring a consistent user experience across different chat modes also required several design iterations.

What did I learn?
- Collaborating on UI design helps align both technical functionality and user experience.
- Even if features are not yet complete, building a solid visual structure makes the next development steps more efficient.

Future Plans:
- Connect the implemented UI elements to the actual server-side logic.
- Finalize and test the chat functionalities across all lobby types.
- Continue working collaboratively to polish and integrate remaining components.

Date: April 4, 2025
Author: Aiysha

What did we do today?
We experimented with several physics libraries (jBox2D, dyn4j) to handle collision detection and gravity. Due to persistent issues with JavaFX module not found errors, we ultimately decided to implement our own collision detection and gravity system.
We restructured our game objects by creating custom classes for players, moving platforms and static objects.
We updated our collision resolution logic so that static objects (like platforms) aren't pushed around by players. We also introduced modularity by adding flags (e.g., movable vs. immovable) so that different game objects can have varying behaviors regarding movement and collisions.

Challenges we faced:

- JavaFX module issues: We encountered errors related to missing JavaFX runtime components, which made it difficult to use external physics libraries.
- Gradle and module-path configuration challenges that further complicated the integration of external libraries.
- Balancing smooth interpolation for moving platforms with the need for efficient, lag-free updates.
- Ensuring that our custom collision detection system correctly handled various object types (players, platforms, boxes) with differing movement and mass properties.

What did I learn:

- How to design a custom collision detection and resolution system tailored to the specific needs of our game.
- Techniques for smooth motion using interpolation (cosine interpolation) to make moving platforms appear fluid.

Future plans:

- Implement gravity based on object mass to give players and boxes more realistic behavior.
- Refine the collision resolution further, ensuring that objects like platforms remain static when they should.
- Add mechanics for grabbing and throwing objects, allowing for more interactive gameplay.
- Explore integrating additional game features (e.g., power-ups, enemy AI) once the core physics and collision systems are stable.
- Continue improving the structure and modularity of game objects for easier maintenance and future expansion.

Date: April 5, 2025
Author: Aiysha

What We Did Today

- Reviewed and reworked our player physics system using vector-based integration (acceleration → velocity → position).
  Experimented with implementing gravity and collision detection for the player locally.
- Investigated issues with jump mechanics, particularly the conditions that enable a jump (using onGround and canJump flags).
- Addressed various synchronization challenges between client and server physics updates.
- Ultimately decided to implement our own collision detection and gravity system to avoid JavaFX module not found errors.

Challenges We Faced

- Integrating physics using vector math and ensuring the correct update order (acceleration, then velocity, then position).
- Handling input correctly so that jump impulses weren’t immediately canceled or interfered with by collision detection.

- Dealing with synchronization issues between client-side and server-side updates.
- Resolving the JavaFX module not found error, which hindered our ability to use built-in JavaFX physics.

What Did I Learn

- How to implement basic physics integration (using acceleration to update velocity and then position).
- The importance of maintaining proper state flags (like onGround and canJump) for accurate jump detection.
- Strategies to troubleshoot and resolve synchronization issues in a networked game environment.
- How to use external libraries (e.g., dyn4j's Vector2) for vector math, even though we eventually opted for our own implementation due to module issues.

Future Plans

- Refine our custom collision detection and gravity system to further reduce lag and jitter.
  Optimize the input handling mechanism to improve the responsiveness of player movements and jumps.
- Test extensively across various game scenarios to ensure consistent behavior.
- Investigate integrating more advanced physics libraries once the JavaFX module issues are resolved or a workaround is found.

Date: April 5, 2025
Author: Aiysha

What We Did Today:

- Box Synchronisation:
  Implemented a system to synchronise the state of boxes across clients using client authority. This ensures that box positions, movements, and interactions remain consistent for all players.
- Grabbing Boxes:
  Developed the functionality to allow players to grab boxes, making them follow the player's movements. This included logic to detect when a box is picked up and to update its state accordingly.
- Throwing Boxes:
  Added mechanics for throwing boxes. This feature calculates and applies throw velocity and direction so that boxes behave realistically when thrown.
- Client Authority Based Synchronisation:
  Adopted a client authority approach where the client controlling a box is responsible for its state updates. This helped in reducing server load and making the interactions feel more responsive.

Challenges We Faced:

- Ensuring that synchronisation of box states was both consistent and low-latency across different clients.
- Handling edge cases when multiple players interacted with the same box, ensuring that the client authority model prevented conflicts.
- Fine-tuning the physics for grabbing and throwing to create a realistic and enjoyable experience.

What I Learned:

- How to implement client-side authority to handle real-time object synchronisation in a multiplayer game.
- The importance of state management when multiple clients can interact with the same game object.
- Techniques for implementing and debugging interactive gameplay mechanics like grabbing and throwing objects.
- Strategies to ensure smooth and consistent user experience in a networked environment.

Future Plans:

- Refinement of Physics:
  Continue to refine the physics behind box interactions, including edge cases and potential conflict resolution when multiple clients interact.
- Extend Gameplay Mechanics:
  Explore additional mechanics such as stacking boxes, collisions, and interactions with other game elements.
- User Feedback & Testing:
  Gather feedback from playtesting sessions to further tweak and optimise the synchronisation and interaction systems.

Date: April 5 2025

Author: William

What We Did Today:

- debugging mainclass to ensure UI Launches successfully.
- Fixed crashes tied to JavaFX initiation orders.
- Writing Projectplan.

Challenges:

- Pinpointing the exact cause of JavaFX launch errors when multiple threads tried accessing GUI elements simultaneously.

What did I learn?

- How to systematically debug a problem.

Future Plans

- implement the rope into the GUI.

Date: April 6/7 2025

Author: William

What We Did Today:

- writing the manual
- writing the project plan with gantt project
- Updating Network Protocol
- Edit README
- implement cs108
- Testing and debugging

Challenges;

- Project Plan and debugging of the main class

Future Plans:

- Unittests
- Rope implementation as an item

Date: April 6 / 7, 2025
Author: Aiysha

What We Did Today:

- **New Game Objects – Key and Door:**
  - Developed a new key object with mechanics similar to the box. The key can be grabbed and thrown, adding to the interactive gameplay.
  - Implemented a door object that acts as a win condition trigger. When the key touches the door, a winning message is sent, marking a successful game completion.
  -
- **Chat Integration:**
  - Integrated the chat features into the main game interface, so players can communicate seamlessly during gameplay.Synchronized chat functionality with other physics-based features, ensuring that both communication and gameplay updates occur in real time.

Challenges We Faced:

- Ensuring smooth integration of the new objects (key and door) into the existing client authority-based synchronisation system.
- Handling the collision detection and interactions between the key and the door to reliably trigger the win condition.
- Merging the new chat functionalities into the main game interface without disrupting the existing physics and synchronisation logic.

What I Learned:

- How to extend existing game mechanics by adding new objects that share similar properties to established ones (like the key with box mechanics).
-

Future Plans:

- Refinement and Balancing:
    - Further refine the interaction mechanics of the key and door, and ensure that collision detection is reliable under various gameplay scenarios.
    - Balance the new win condition and explore additional gameplay events triggered by object interactions.
- Feature Enhancements:
    - Expand chat features to include more advanced functionalities like private messaging, chat history, and improved UI responsiveness.
- Optimization:
    - Optimize synchronisation and performance of both the physics and chat systems to maintain smooth gameplay in larger multiplayer sessions.
- User Testing:
    - Conduct user testing to gather feedback on the new objects and integrated chat features, then iterate on the design based on the insights gained.