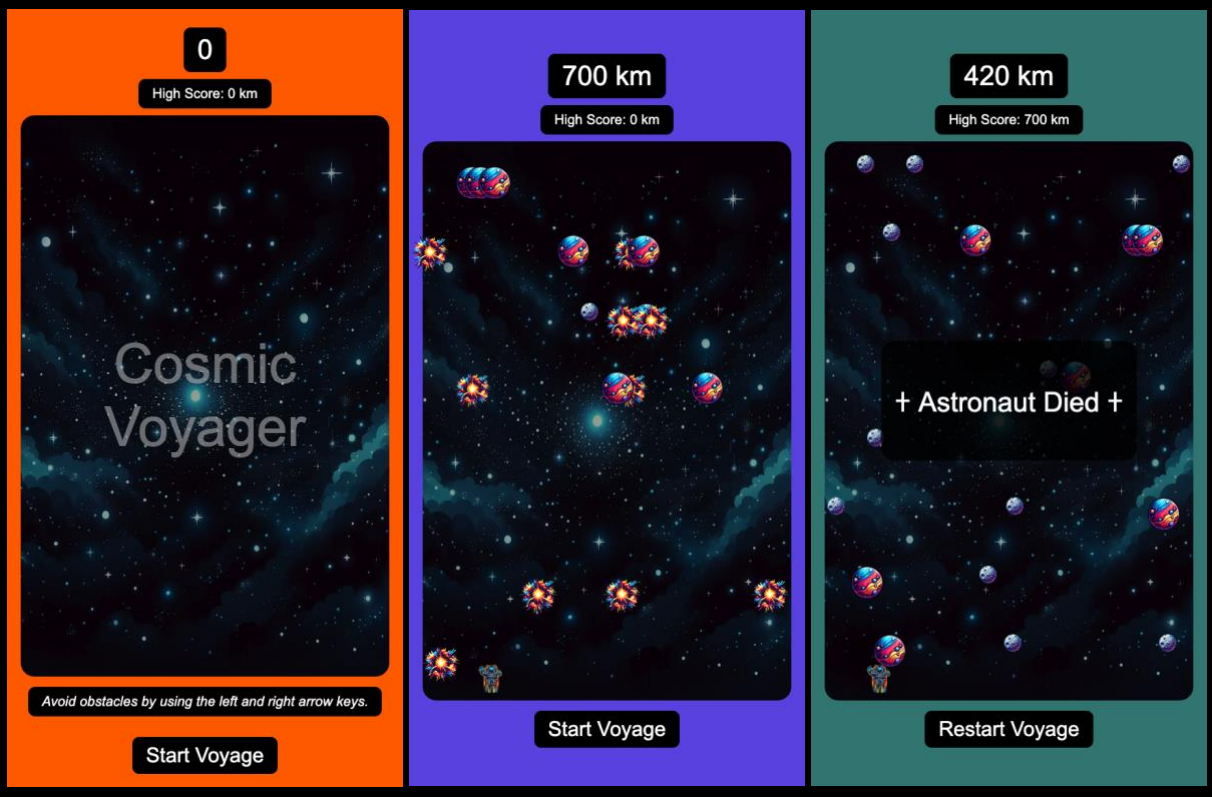
The Creation of Cosmic Voyager: A Game Design Journey

Introduction and Motivation

I wanted to improve my skills in web development by working on a cool and engaging project that I could showcase to my friends and family. I wanted something that would be both a fun learning experience for myself and entertaining for others. With that in mind, I decided to implement a simple yet fun game that would allow me to apply my web development skills while creating something enjoyable for others to play.

This is the result – My web game *Cosmic Voyager*:



Game Design

The first step in creating *Cosmic Voyager* was to decide on the type of game I wanted to develop. I specifically chose to implement a browser-based web game because every player could easily play it on multiple different devices without having to download an app. I envisioned a simple, retro-inspired 2D web game akin to classic Atari games. I wanted something that evoked a sense of nostalgia but was accessible to anyone. The

gameplay had to be straightforward but engaging: You control an agent, avoiding obstacles that move from the top of the screen towards the bottom. The longer you survive, the higher your score climbs. The objective was to create a game that anyone could pick up easily but would still offer a satisfying challenge.

To develop an interesting challenge progression, I began with small, manageable obstacles that were easy to avoid. As players advance, they encounter larger obstacles at key score checkpoints, which increases the difficulty in a natural and incremental way. The difficulty escalates further with obstacles that shift horizontally and obstacles that increase in size as they descend. I also implemented a randomizer to determine how many obstacles are generated each second and their positions, which ensured that no two play sessions felt exactly the same. I wanted to keep the player constantly adapting to new challenges, and this system created a sense of excitement as the difficulty ramped up. This kept the gameplay fresh and progressively challenging, without making the game feel impossible.

The theme of the game was crucial in creating the right atmosphere. I chose an astronaut and universe theme to give the game a cosmic flair and a sense of exploration. The playable agent is represented by an astronaut, while the obstacles are themed accordingly: smaller obstacles are asteroids, larger obstacles are planets, horizontally moving obstacles are supernovas, and the growing obstacles are black holes. I wanted players to feel as though they were guiding an astronaut on a perilous journey through space. I used DALL-E 3 to generate icons for the agent, obstacles, and backgrounds, capturing that cosmic feel. After generating the assets, I optimized them with image editing software to achieve the exact look I desired—balancing vivid colors with a sense of interstellar wonder. The result was a polished set of game elements that enhanced the player's immersion in the game world.

Game Mechanics

The core mechanics were designed to provide a simple yet thrilling experience. The player's primary task was to navigate the astronaut through the barrage of obstacles. Players could control the astronaut either with the keyboard or through touch controls for mobile devices, which made the game accessible across different platforms. It was important for me to make the game playable on both desktop and mobile so that as many players as possible could enjoy it.

The collision mechanics ensured that players had to make precise movements to survive. Each moment required full attention, as a single collision would end the voyage. The game rewards players through a gradually increasing score, providing a simple but effective incentive to stay alive as long as possible. I implemented a score counter to display the current run's score and also a session high score, which added a layer of motivation for players to continually improve. When the astronaut hits an obstacle, the

game ends, and a "Astronaut Died" Game Over screen is displayed, adding a dramatic conclusion to each playthrough.

Mechanics for obstacle movement were carefully thought out. Initially, obstacles moved vertically at a consistent speed, creating a baseline challenge. As the game progresses, however, new types of obstacles appear. Some obstacles begin to oscillate horizontally, forcing players to think ahead and time their movements more precisely. Other obstacles grow larger as they move downward, demanding even more careful navigation. Balancing these mechanics was crucial to creating a game that was challenging yet fair. I wanted players to feel a sense of accomplishment when they successfully dodged a particularly tricky obstacle.

To maintain variety and excitement, I also added visual and gameplay cues that indicated increasing difficulty, such as changes in background color when certain score thresholds were reached. This signaled progress to the player and added a layer of dynamism to the experience, making each new phase of the game distinct from the last.

Implementation

With the mechanics and design established, it was time to implement the game. I used HTML for the game's structure, CSS for styling, and JavaScript for the game logic that made *Cosmic Voyager* come to life. This development process was filled with both triumphs and challenges. One of the biggest challenges was optimizing the game for different screen sizes—making sure that it worked well on everything from large desktop monitors to small smartphone screens. This required a lot of careful adjustments, especially to ensure that touch controls felt as intuitive as keyboard controls.

The visual design of the game also required thoughtful CSS styling to keep the game visually appealing while maintaining simplicity. Elements like the astronaut, obstacles, and the game area were styled to ensure a cohesive cosmic theme, using a consistent color palette and rounded borders to evoke a playful yet space-themed atmosphere. I paid particular attention to detail in designing the astronaut's movement and the look of each obstacle. Dynamic elements, such as the changing background color when players reached specific score milestones, were added to enhance the game's sense of progression and keep players engaged over time.

Balancing gameplay elements also proved to be a critical aspect of implementation. The JavaScript code governing the movement of the obstacles, astronaut controls, and collision detection had to be tuned carefully to create a fluid and responsive gameplay experience. I wanted to make sure that the astronaut's movement felt smooth and that players had a fair chance of avoiding obstacles if they reacted in time. This meant adjusting things like obstacle speed, frequency, and movement patterns until they all felt just right.

Implementation Duration

A significant part of the success of *Cosmic Voyager* was the speed at which it was developed, thanks in large part to generative AI tools like ChatGPT and DALL-E 3. The whole game code, including all files, amounts to roughly 600 lines of code. It took me roughly 10 hours to complete the game design, which included brainstorming the mechanics, sketching out the gameplay flow, and creating the visual elements like icons and backgrounds. This was made possible by leveraging ChatGPT for brainstorming and rapid iteration, and using DALL-E 3 for generating graphics, even though I don't have a professional background in game design. The support of these AI tools significantly reduced the amount of time I needed to spend on concept design, which allowed me to move quickly into implementation.

The implementation phase, including coding the HTML structure, CSS styling, and JavaScript logic, took approximately 20 hours. This included writing the core game loop, implementing the movement mechanics, and creating the collision detection system. ChatGPT played an instrumental role in guiding me through various programming challenges, such as writing efficient JavaScript for the obstacle generation and refining the collision detection logic. While ChatGPT did most of the coding and provided crucial support in solving issues, it was essential for me to understand the underlying logic myself in order to effectively guide ChatGPT in the right direction, particularly by providing feedback on the game's behavior during playtesting. Whenever I encountered issues—such as making sure that the game scaled properly to different devices—I could ask ChatGPT for advice, which saved me countless hours of trial and error.

Fine-tuning parameters such as the obstacle speed, the astronaut's movement responsiveness, and the frequency of new obstacles required several iterations to get just right. I wanted to make sure the game was challenging but not overly punishing, and this meant testing and adjusting numerous variables. The fine-tuning process took additional time, but it was essential in making the game playable and fun. I also focused on the pacing—ensuring that the challenge ramped up in an exciting way that kept players on their toes without becoming frustrating.

Testing & Deployment

After implementing the core game mechanics and completing the initial tuning of game parameters, I deployed *Cosmic Voyager* on GitHub and made it available to friends for playtesting. Their feedback was invaluable in identifying areas where the game could be improved. For instance, some players found certain phases too difficult, while others suggested tweaks to the astronaut's responsiveness. Based on this feedback, I optimized several parameters, such as the speed and size of obstacles, to enhance the overall playability.

I also implemented a randomizer to determine how many obstacles are generated each second and their positions, ensuring a unique experience every time someone played. This randomness added an element of unpredictability that kept players engaged. Additionally, I tested the game in multiple browsers, including Chrome, Firefox, and Safari, to ensure that it performed consistently across different environments and devices.

Next Steps

Looking ahead, I want to take *Cosmic Voyager* to the next level by using it as a platform to train an agent with Deep Reinforcement Learning. My goal is to develop an AI that can learn to play the game autonomously, navigating the astronaut through increasingly difficult obstacles while maximizing its score. This will involve setting up a training environment, defining appropriate rewards for the AI, and iterating on the training process to improve its performance. I believe this project will be an exciting exploration into the capabilities of AI in gameplay optimization, and it will be fascinating to see how well an AI can master the challenges of *Cosmic Voyager*.

Conclusion

Cosmic Voyager represents what's possible when creativity meets powerful AI tools. In just a couple of days, I was able to design, code, and polish a fully functional web game that delivers a fun, retro-inspired experience. The combination of simple but effective gameplay, an engaging astronaut theme, and a steadily increasing difficulty curve makes *Cosmic Voyager* an exciting challenge for players of all ages. The process of creating this game was both rewarding and educational, demonstrating how much can be accomplished even without a formal background in game development.
Throughout this project, I experienced all the key phases of the software development life cycle: from design, to implementation, to testing, to deployment, and finally to maintenance. Each step was essential in bringing *Cosmic Voyager* to life and ensuring that it provided a smooth and enjoyable experience for players. The iterative process of design and refinement exemplifies the importance of thorough testing, deployment for user feedback, and continuous improvement in software development.

If you're interested in diving deeper into the code, you can find my GitHub repository linked [here]. I hope you enjoy playing *Cosmic Voyager* as much as I enjoyed creating it!

You can play the game yourself and try to achieve a score above 2000 or try the AI mode right [here].

Thanks and have fun! Best, Andy