Generative Art

Task 2 & 3

Arianne Bezzina

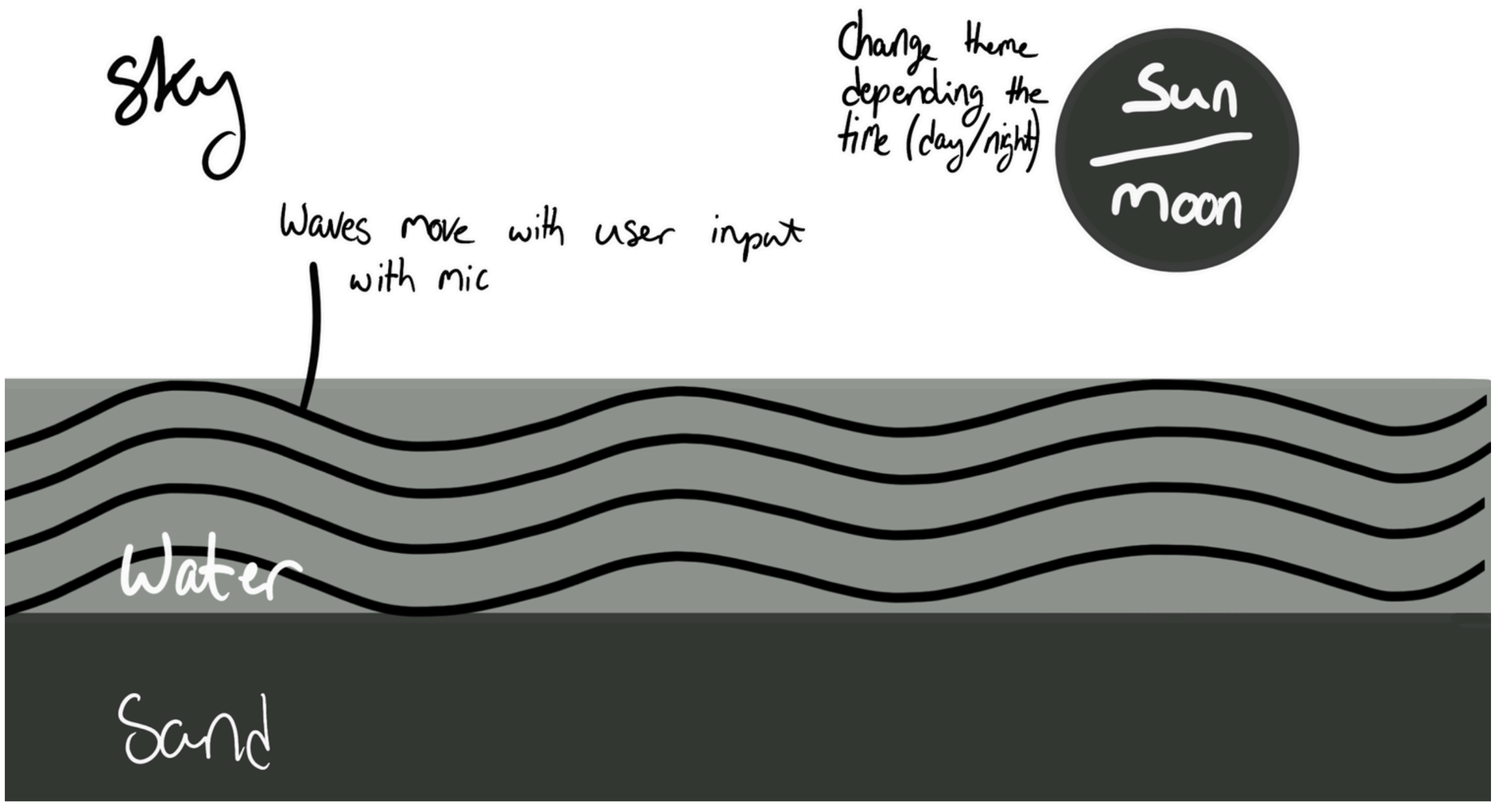
Project Description

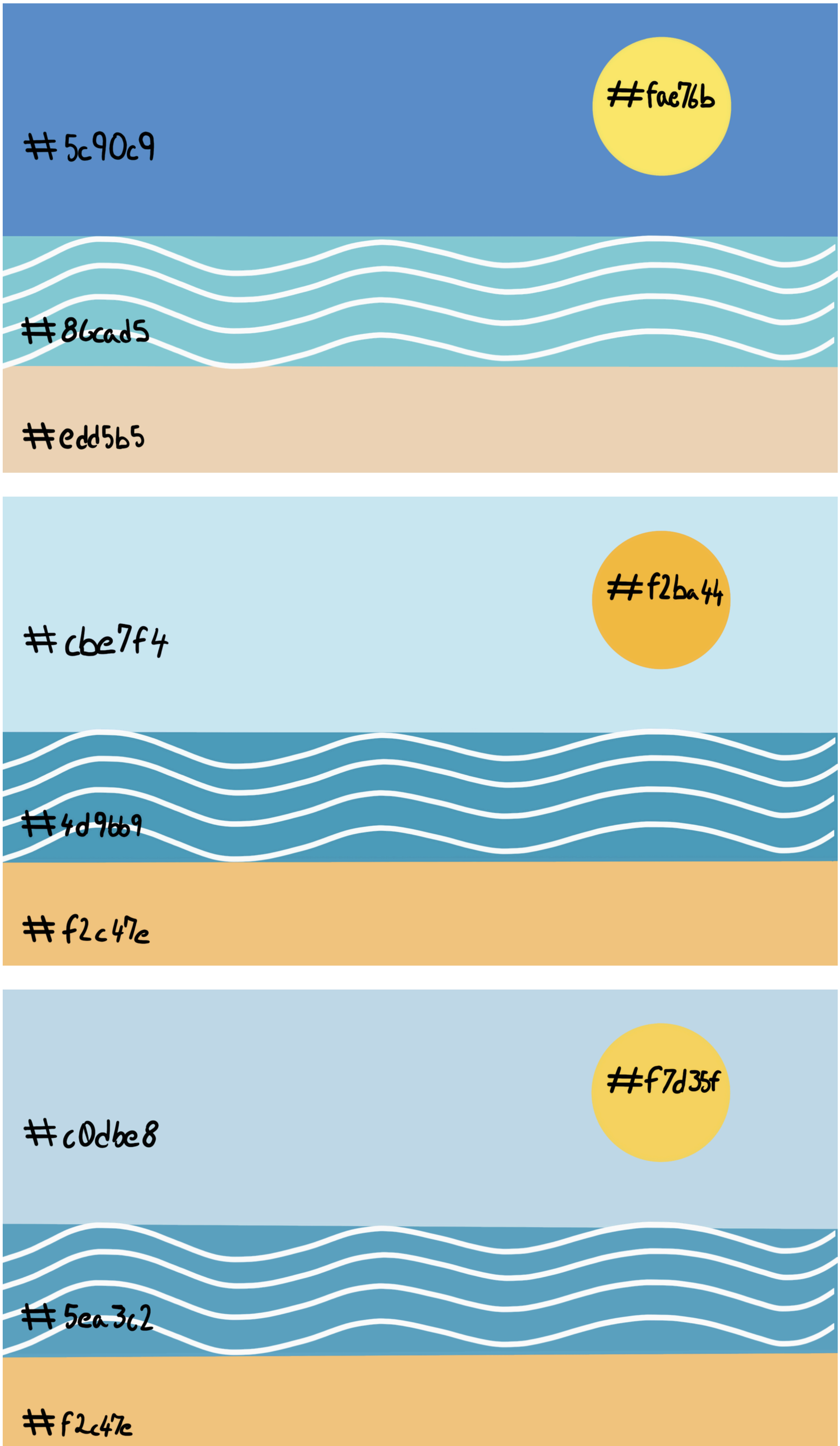
I will create an interactive generative beach scene banner using P5.js. The banner will feature a dynamic sky, a sun or moon, stars, and animated waves. It will automatically change themes between sunrise, day, and night based on the user's real-time system time. The waves will react to real-time microphone input, allowing users to interact with the scene through sound.

Generative Techniques Used

The techniques I used include **randomness** for generating star positions and **noise** to create smooth, natural wave motion. I also incorporate real-time data using the getHours() function to switch between sunrise, day, and night themes based on the system time. Additionally, the banner responds to microphone input using the getLevel() function from p5.AudioIn(). When the user speaks or plays music, the wave height increases through a **live data stream.**

Design




#5c90c9
#fae76b
#86cad5
#edd5b5
#cbe7f4
#f2ba44
#4d9bb9
#f2c47e
#c0dbe8
#f7d35f
#5ea3c2
#f2c47e


#132a48
#1a4766
#7c6144


#112436
#475c6f
#7c6144

Challenges

One challenge I faced was changing the background colour of the sea during night mode. Initially, I had used a hardcoded fill colour directly inside the drawWaves() function, which meant the sea background stayed light blue all the time, even at night. To fix this, I updated the getThemeColors() function by adding a new property called waveFill. This allowed me to define different sea colours for day and night. Then, in the drawWaves() function, I replaced the hardcoded colour with theme.waveFill, which made the background colour change correctly depending on the time of day.

```
//Waves move based on mic input
function drawWaves() {
  let volume = mic.getLevel();
  let waveHeight = map(volume, 0, 0.3, 10, 100);

  //Draw filled base water (light blue)
  noStroke();
  fill('#ADD8E6'); // Light blue fill
  rect(0, height - 80, width, 40);
```

```
//Draw filled base wat
noStroke();
fill(theme.waveFill);
rect(0, height - 80, w
```

At first, the waves weren't reaching the end of the canvas. I was using < width in the for loop, but after some research, I realized the issue and updated it to <= width. This fixed the problem and allowed the waves to cover the full width of the screen.

```
for (let x = 0; x < width; x += 10) {
  let noiseVal = noise(x * 0.01, y * 0.01,
```

```
for (let x = 0; x <= width; x += 10) { // include the
full width
```

The waves were going too high due to the sound input values. To fix this, I added an if condition to limit the maximum wave height, ensuring the animation stays within a controlled range.

```
//Waves move based on mic input
function drawWaves() {
  let volume = mic.getLevel();
  let waveHeight = map(volume, 0, 0.3, 10, 100);

  //Draw filled base water
  noStroke();
  fill(theme.waveFill);
  rect(0, height - 80, width, 40);

  noFill();
  stroke(theme.wave);
  strokeWeight(2);
```

```
//Waves move based on mic input
function drawWaves() {
  let volume = mic.getLevel();
  let waveHeight = map(volume, 0, 0.3, 10, 100);

  if (waveHeight > 30) {
  waveHeight = 30;
  }
```

Debugging techniques

To debug the wave height issue caused by microphone input, I used console.log() to monitor the volume and waveHeight values in real time. This helped me decide the appropriate value to use in the limiting condition to prevent waves from going too high.

While testing the theme colour changes for sunrise, day, and night, I temporarily replaced the getHours() function with let hour = 6 so I could manually test different times and confirm that the theme switched correctly.

When running the P5.js sketch in the browser, I encountered an issue where the microphone input was not functioning correctly, and the volume values were stuck. To fix this, I added userStartAudio(); in the setup() function, which allows audio input to work after a user interaction, as required by modern browsers.

Contemporary Media Influence

My generative art banner makes websites more interactive and data-driven by using real-time inputs like microphone sound and system time. It can be used in music visualizers, website banners, or art installations to create a more appealing visual experience and increase user engagement through live interaction.

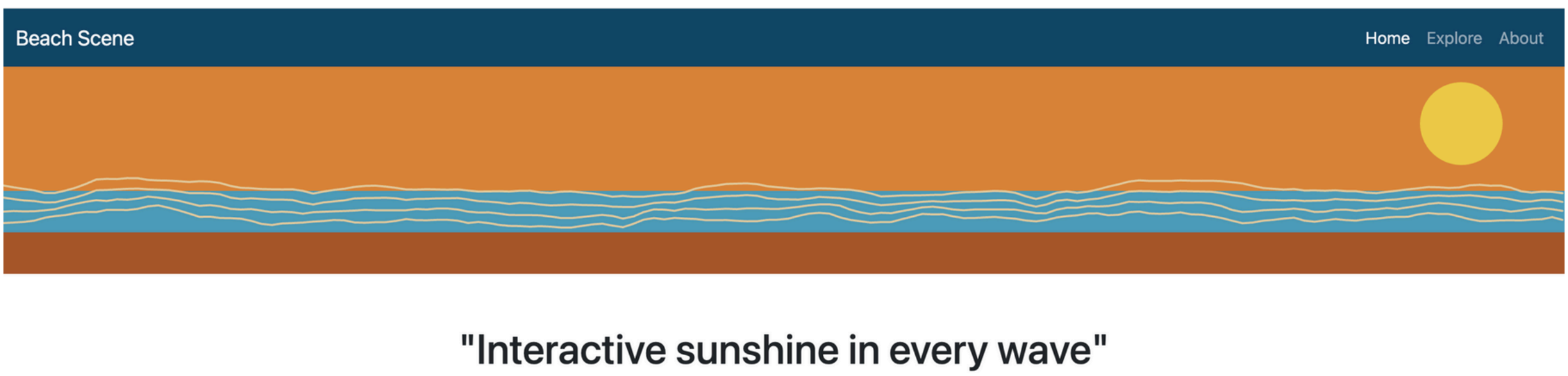
Future Implications

Generative art, as shown in my project, suggests that future creative practices will include more automation, interactivity, and personalization. Artists can use code as a creative tool, combining logic with visual design. Projects like mine show how digital artworks can evolve over time, respond to the environment, and be customized by users. This can influence areas such as UX design, live performances, and interactive learning tools.

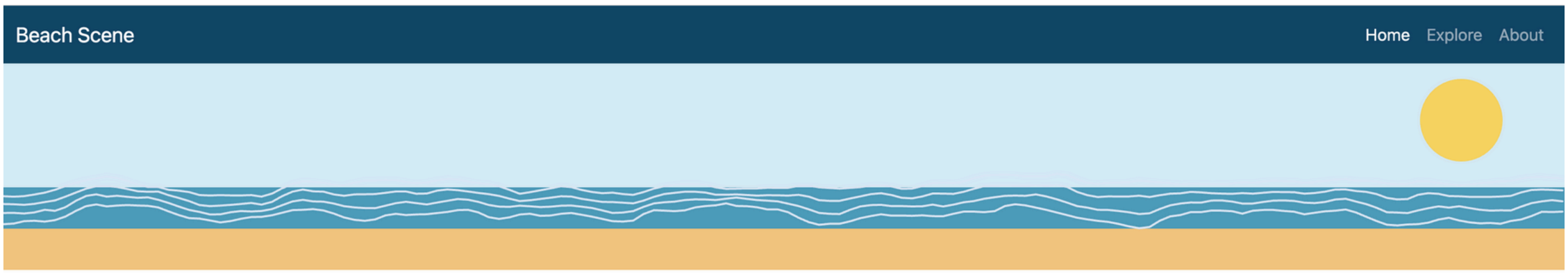
Reflection

I’m happy that I learned how to use P5.js and implement it into HTML, as this was something I didn’t know how to do before. I’ve always enjoyed both coding and art, so combining them through generative design was an exciting and rewarding experience. If I had more time, I would have added more detail to the beach scene and explored additional elements to make the animation even richer and more immersive.

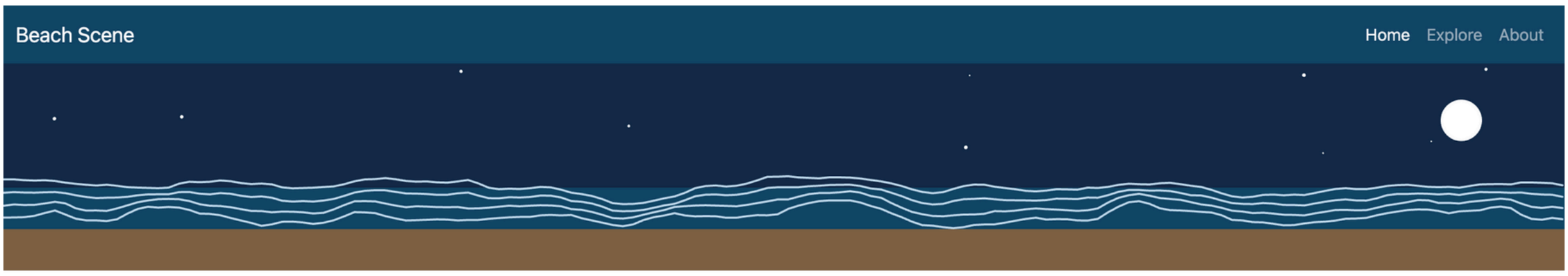
Final Banner



"Interactive sunshine in every wave"



"Interactive sunshine in every wave"



"Interactive sunshine in every wave"