Final Project

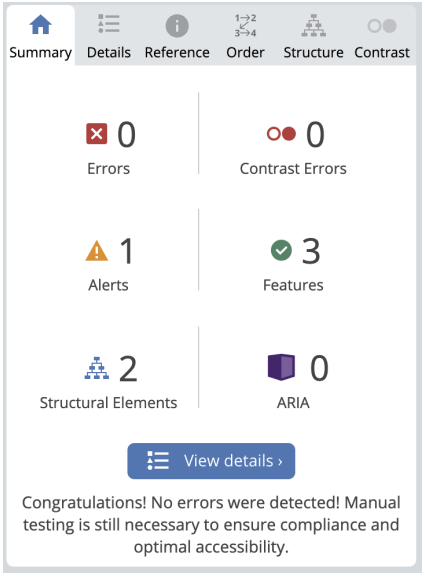
Programming Usable Interfaces - Adrian Schmidt (agschmid)

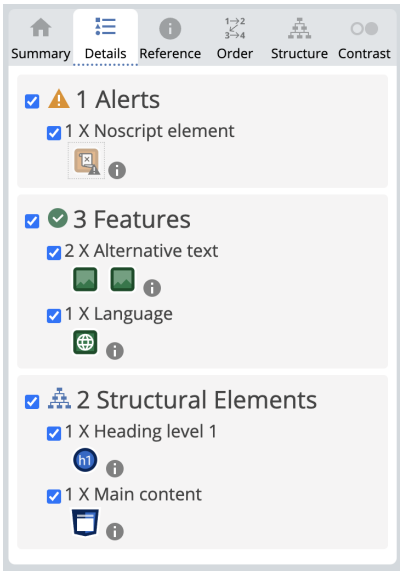
Final Website: https://agschmid.github.io/final-project/ping/

Screen Sizes: The project should work for screens larger than an iPhone SE (375px x 667px), but I mostly tested on an iPhone 13 Pro Max (430px x 932px) and my Macbook (1512px x 800px).

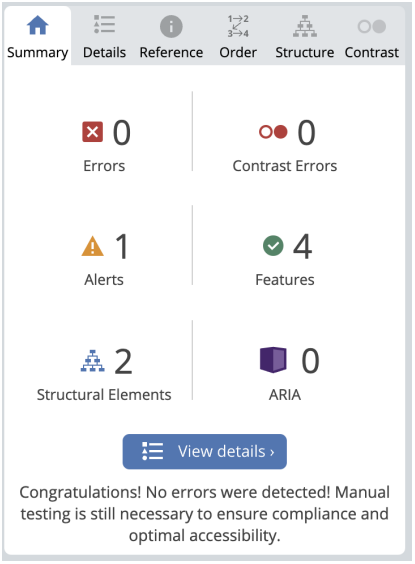
Accessibility Screenshots:

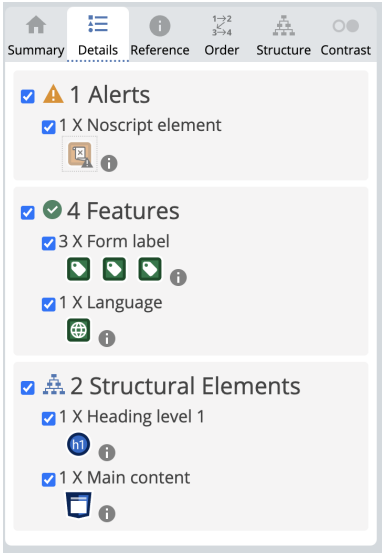
Homepage:



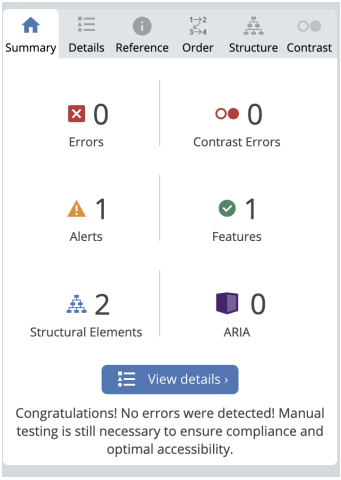


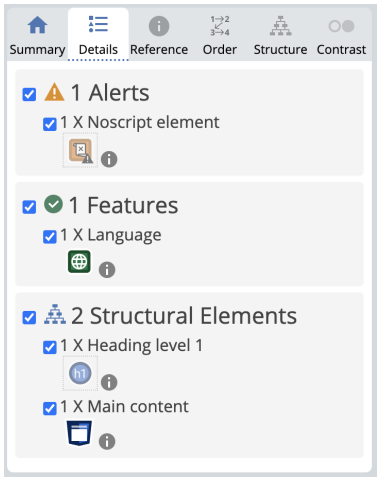
Accessibility:



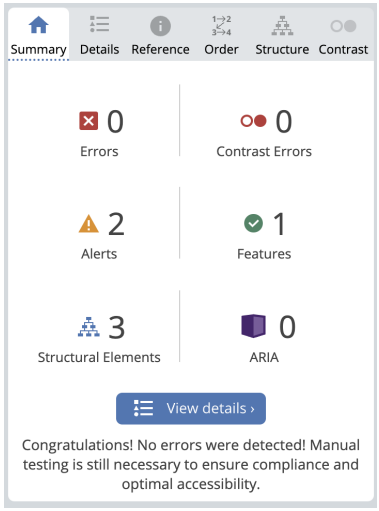


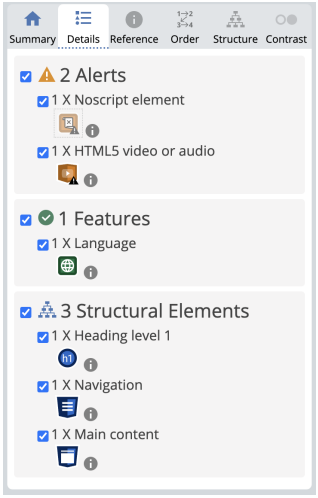
Countdown:



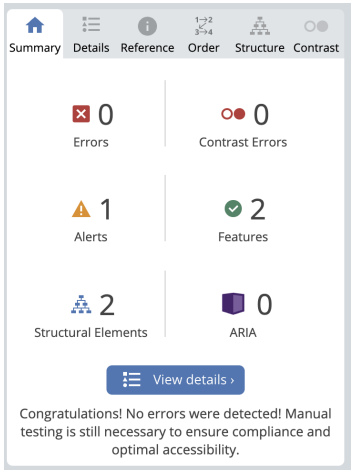


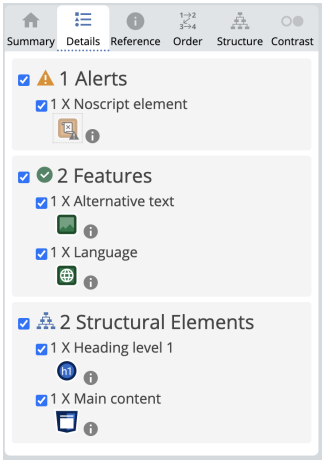
How To Play:



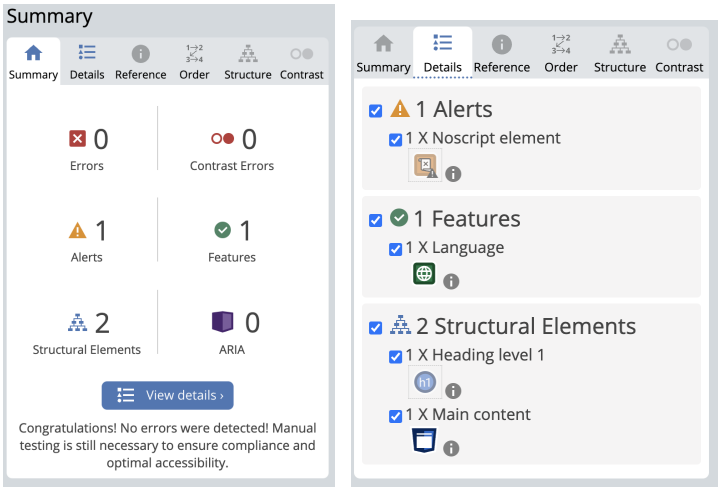


Paused Screen:

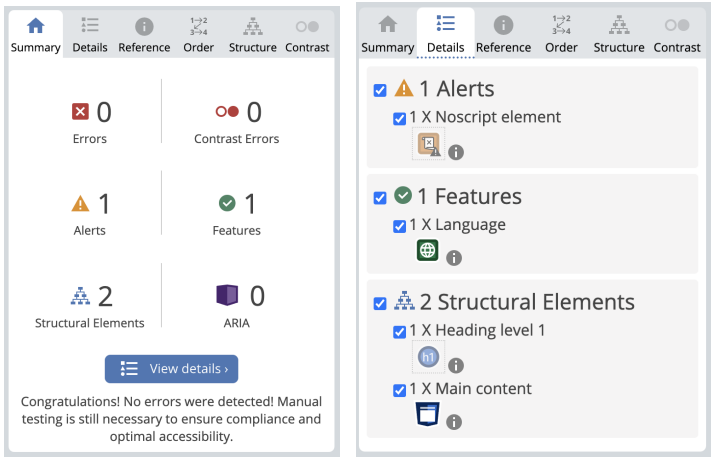




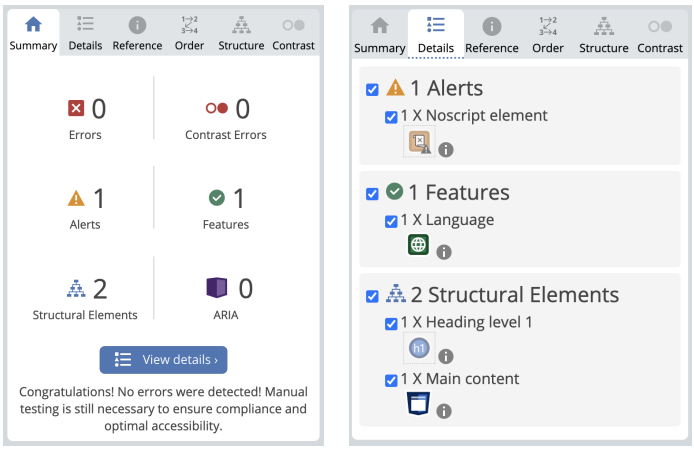
Exit Confirmation:



Game over:



Playing



Part 1: Description of my website

My website is a 3D Pong-style game played in the browser on mobile phones or computers. The audience is people who want to play a quick game with an increasing difficulty curve. I wanted the game to be challenging, as this is a crucial factor in the replayability of high-score games like Flappy Bird.

It has a 'synthwave' aesthetic, combining the styles of 80s films like Tron with early computer visuals like those used in Windows 2000. The game is played by moving a blue paddle around the screen with either the mouse or touch controls. The ball will bounce between the player's paddle and the enemy's paddle, with the score increasing each time you hit the ball. The ball also speeds up each time you hit it, increasing the game's difficulty.

The website also has dynamic sounds, with a seamlessly looping music track and feedback for when the player hits the ball and when it hits the edges of the screen. There is also visual feedback in the paddle, letting the player know when they hit the ball, and also in the edges of the screen to indicate the ball's location to the player. The game begins with a tutorial the first time you play, which can be re-accessed through the main menu.

As accessibility was a focus for this course, I wanted to include features allowing more people to play the game. Features include compatibility with many screen sizes, the ability to increase the size of the paddle, and adjusting the game's speed. With more time, I would have also included keyboard controls for the game. However, the user can also control each menu using tab selection. Finally, all accessibility options and the high score are kept in localStorage, meaning the user doesn't have to re-enter them.

Part 2: Interacting with the website

The primary interaction is playing the game, but there are a variety of menu options to modify the experience.

- Access different menus by clicking/tapping on the different buttons. These can also be selected accessibly using the ‘Tab’ key and pressing ‘enter’.
- Enable music by pressing the ‘play music’ button. Stop the music by pressing the button again.
- In the accessibility menu, click/tap and drag sliders to adjust settings. The sliders can also be adjusted using tab selection and the arrow keys.
- To increase/decrease the speed of the game, adjust the speed increase slider.
- To change the size of the paddle, adjust the paddle size slider.
- To adjust the volume of the music, adjust the music volume slider.
- Start the game by clicking/tapping the green play button on the home screen.

- For the web, play the game by moving the mouse pointer around the screen. This controls the blue ‘paddle’, which can bounce the ball back towards the enemy.
- For mobile, play the game by tapping and dragging around the screen. This controls the blue ‘paddle’, which can bounce the ball back towards the enemy.
- Keep adjusting the position of the paddle to hit the ball back and forth. If you miss, the game over screen will show. Quickly start a new game by pressing ‘Play Again’, or go back to the home screen by pressing ‘Exit Game’.
- To pause the game, click the pause button. Alternatively, press the escape key. The escape key is indicated on the pause button on the web, but not on mobile.
- When playing the game on the computer, dragging the mouse out of the central area makes it visible again, so you can more easily click the pause button.
- All the accessibility and music can still be adjusted in the pause menu, by clicking the ‘accessibility’ button.
- To exit the game from the pause menu, click the red Exit Game button, and then confirm by clicking ‘Yes’, or return by clicking ‘No’. This second layer ensures a player doesn’t accidentally quit a paused game.
- The tutorial is shown before playing a game if the high score is 0. It can be reaccessed by clicking the ‘how to play?’ button in the home menu.
- All the accessibility settings and the high score are stored locally. Refresh the page to see that the settings and score carry over.

Part 3: External tools

Name of tool: React Three Fiber
Why did I choose to use it? I started with three.js, as it is the primary 3D web library, allowing me to program visuals for the game more easily. I quickly realized I also needed to have dynamic menus and states. React Three Fiber enables three.js canvases to be rendered with React, meeting my needs.
How did I use it? I used React Three Fiber to create all the 3D elements of the page as React functions. These elements include the ball, the paddles, and the grid, which is dynamically lit based on the ball's position. In addition, I used the postprocessing library to add 'Bloom' to the page, giving it a more hazy effect.
What does it add to my website? React Three Fiber is the backbone of my website – it allows me to build all the 3D elements and place and adjust the lighting. Additionally, it allowed me to have a frame loop, where I could consistently run my calculations to change the ball's position appropriately.

Name of tool: Howler
Why did I choose to use it? Howler is a simple audio library that allows me to play sounds more quickly through the Web Audio API, allowing for looping and transitioning between sounds without a gap (which is not typically possible with HTML audio files).
How did I use it? I use Howler to play all sound effects and music, including whenever the player or enemy hits the ball or the ball bounces. The immediate transition between sounds meant I could create an intro section of music that seamlessly transitions into a looping one.
What does it add to my website? The background music adds more fun and intensity to the game, while the other sound effects provide helpful feedback to the player on the ball's position.

Name of tool: Davinci Resolve
Why did I choose to use it? I wanted to edit videos/audio, and it's a free and industry-standard program.
How did I use it? I used Davinci Resolve to edit my tutorial videos so that they fade and loop less jarringly than if they had a hard cut between the start and end of clips. I also used it to edit my sounds so the music could loop seamlessly, and the hits had reverb.
What does it add to my website? It allowed me to add polish to the video and audio elements, ensuring there wouldn't be sudden changes when those elements looped.

Name of tool: Adobe Illustrator
Why did I choose to use it? I needed to create textures for the game, and Illustrator is the industry standard program.
How did I use it? I created the paddle textures, with the glowing effect, in Illustrator. I also made the custom slider image as an SVG file in Illustrator.
What does it add to my website? Illustrator allowed me to make textures more aligned with my design visions for the website.

Name of tool: Mobile Detection
Why did I choose to use it? As I wanted the game to work on both mobile and desktop, I knew that some react elements would need to be conditionally rendered based on the type of device and, therefore, input.
How did I use it? I used it to dynamically change instructions on the website (swapping stuff like dragging the mouse with dragging your finger). I also adjusted the pause button to reference the escape key on desktop but not mobile.
What does it add to my website? It makes the website feel like it belongs on the user's device, whether mobile or desktop. Knowing this is important for usability, as the user may think they shouldn't be using the website on that device if they see conflicting information.

Part 4: Prototype Iteration

After showing the prototype to the class in the bonus session, I added in more designs to make the page easier to use, including key presses to access the menu, and a tutorial at the beginning. I had originally planned for the game to end by making the enemy miss, but my discussions during the critique pushed me towards a system where the goal was just to hit the ball as many times as possible. I had one prototype where the accelerometer and phone rotation was used to control the ball, however the control scheme did not feel very intuitive, so I opted to remove it. I also updated the grid from being a static texture to a series of rendered lines, allowing for dynamic lighting changes based on the state of the game. I added an accessibility menu, to try and allow for people with different abilities to still play the game – with further time I would have also included keyboard controls to allow for additional control types. Based on the final presentation session, I incorporated feedback about font-sizes for the accessibility screen (increasing legibility of the content), and styling consistencies (increasing the cohesiveness and satisfaction of the design).

Part 5: Challenges

My first big challenge was having a single loop for rendering, as many react components change the state, and I didn't want to affect these by passing many functions between components. My solution for this was to have a single react component that contains hooks accessing the state of all the other components. Another challenge was getting mobile controls to work well, as I initially wanted the game to be controlled using the accelerometer, but once I had this working, I realized it was a worse experience than using touch. Finally, the physics calculations took quite some effort, including calculating whether the ball hit the paddle, the velocity changes when the ball bounces, and transforming between pixel coordinates to the react-three-fiber coordinate system.