Politico – System Design Documentation

Joshua Jackson – P16179167

Introduction

This document will cover the system design for my final year project. I will be documenting the architectural approach I have taken as well as how that looks in the real-world with a discussion on class/component design alongside UML diagrams where applicable. Finally, I will show mockups of the user interface as well as a brief discussion around how certain parts of the designs can be made as React components.

Architectural Approach

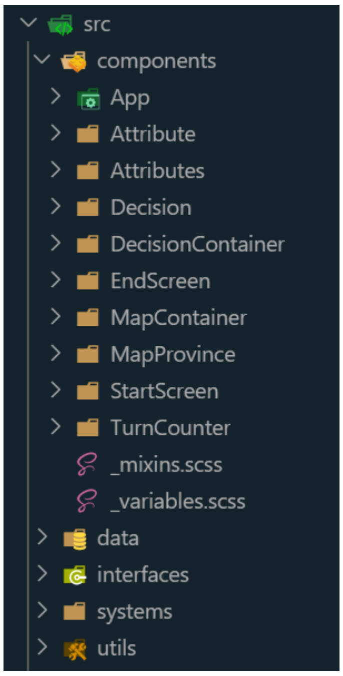
The architectural approach taken is heavily based around having a component driven application where each piece of the UI is split into composable components that are initialized from within a root App component, allowing each piece of the UI to be manipulated individually without requiring a knowledge of the entire application; To aid with this, "Pure" components are used as much as possible which means that the component has no internal functionality, it only displays HTML onto the page and the dynamic part of the component is achieved by passing properties into it, this results in easily testable and maintainable components that don't act as black boxes. Components are kept within their own folder that contains a .TSX file (React) and a .SCSS file (Styling) and every component follows the same structure, where a component has a test, a .test file will also exist keeping a component truly independent from other components.

Where a component isn't pure and is linked to another part of the application, TypeScript classes are used to make sure that connection isn't just obvious to the developer but it also obvious to the compiler.

Functionality that utilizes an API are abstracted away via the use of API-agnostic TypeScript interfaces that make it easy to switch to another API if required without ever needing to update the code that uses the API. This is done by having a Systems class that initialises the API-agnostic systems that, if required, can just be swapped out in only that location. An example of this is the DataStorage class, it is an interface that only has get/set functionality for key/value pairs, the game uses the DataStorage interface but the Systems class intialises this as a browser LocalStorage implementation of that interface. If the game was to use cloud storage, all that would need to happen would be to implement a class with get/set functionality that calls out to the cloud storage provider under the hood without requiring any changes to existing code that relies on DataStorage.

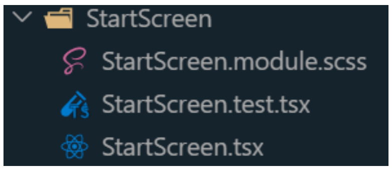
Finally, wherever non component-specific functionality can be abstracted out into a reusable utility, the opportunity is taken as it aids with unit testing and keeping everything maintable.

Folder Structure:



```
src
  components
    App
    Attribute
    Attributes
    Decision
    DecisionContainer
    EndScreen
    MapContainer
    MapProvince
    StartScreen
    TurnCounter
    _mixins.scss
    _variables.scss
  data
  interfaces
  systems
  utils
```

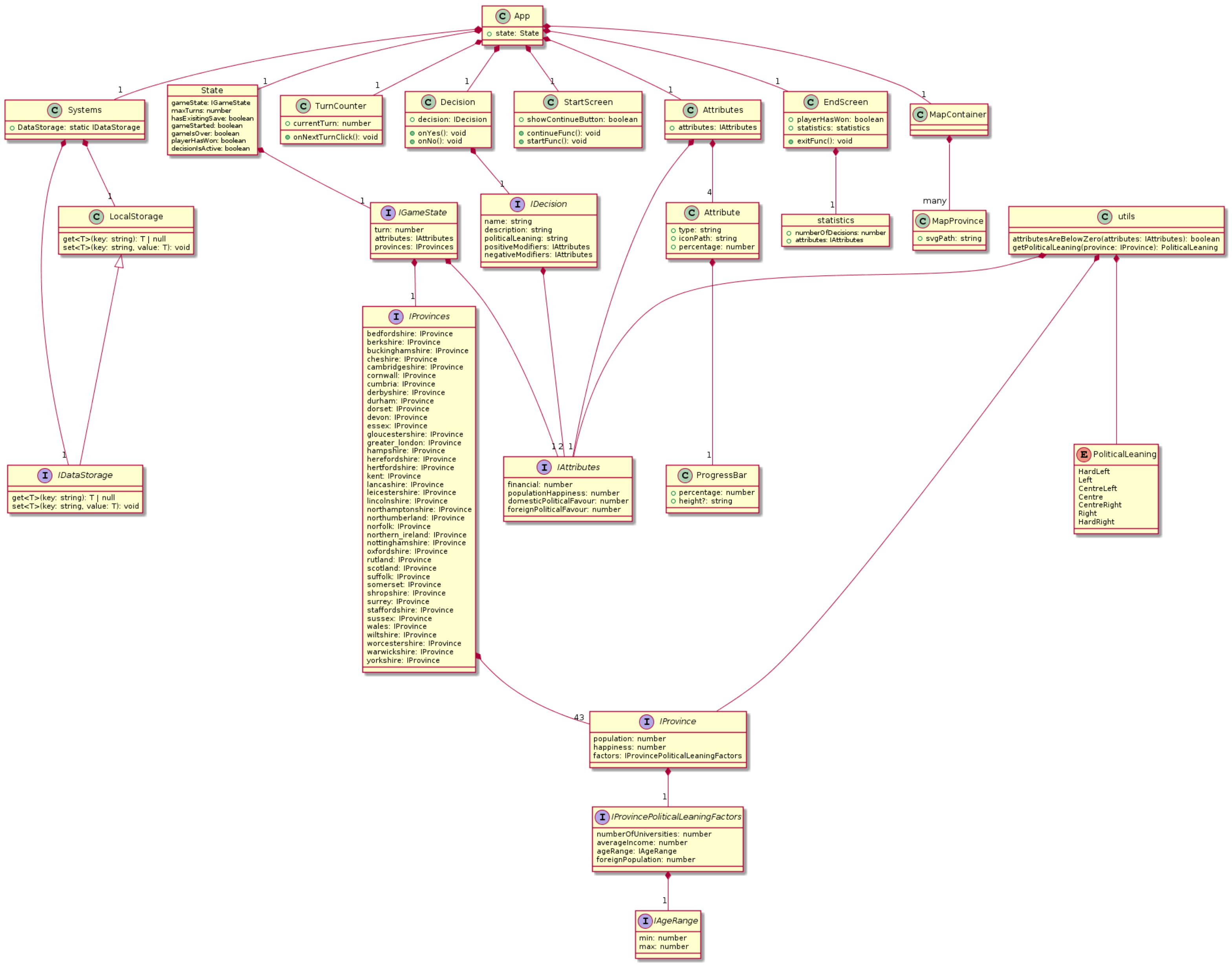
Component Folder Structure:



```
StartScreen
  StartScreen.module.scss
  StartScreen.test.tsx
  StartScreen.tsx
```

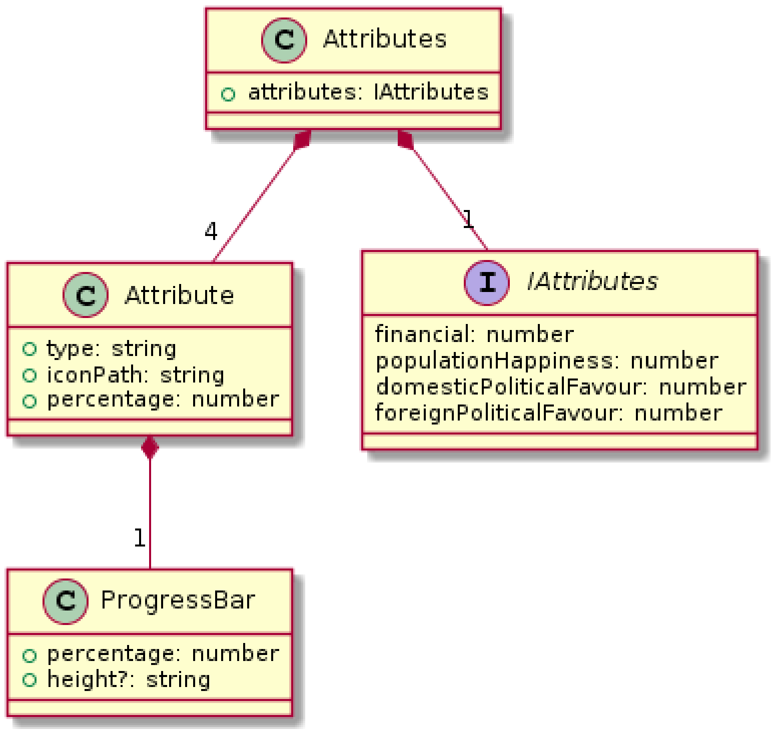
Class/Component Design

App

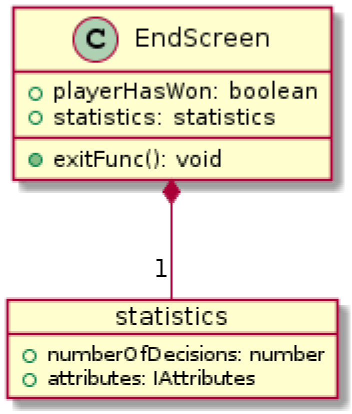


View the above PlantUML diagram by clicking here or view the diagrams of the individual components (where applicable) below.

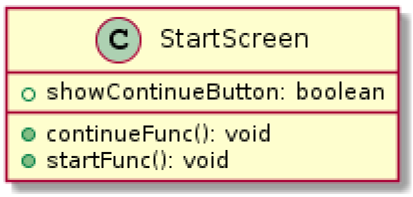
Attributes, Attribute and ProgressBar



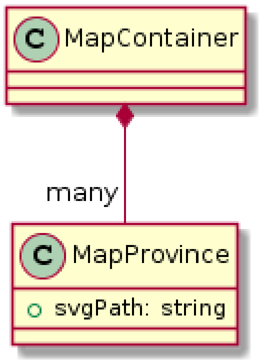
EndScreen



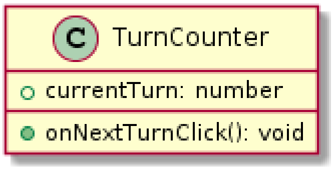
StartScreen



MapContainer and MapProvince



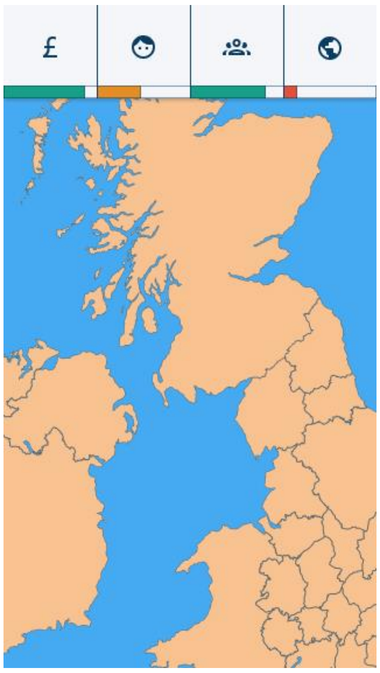
TurnCounter

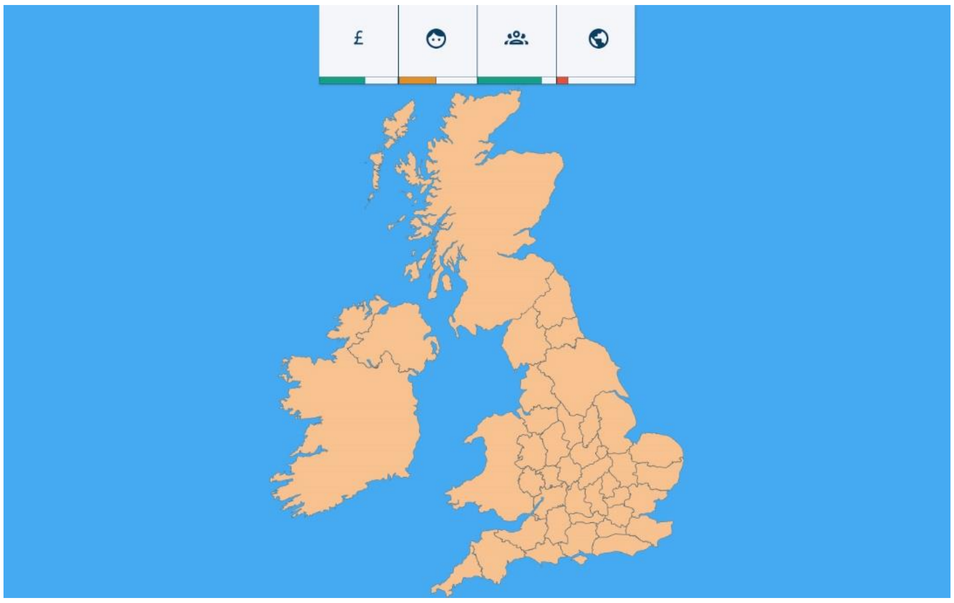


UI Designs

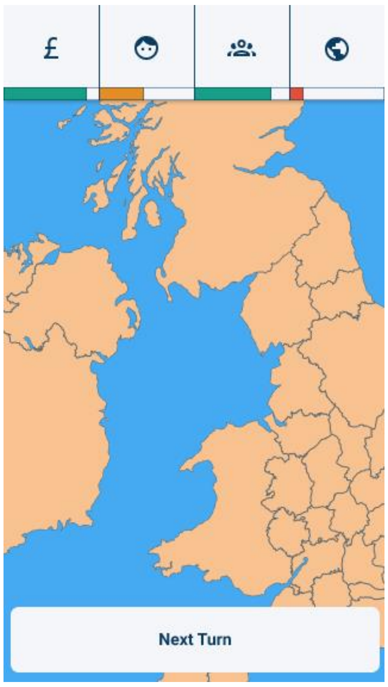
The UI has been designed within the web application Figma with a mobile-first focus that includes both a mobile and a desktop mockup for each piece of functionality.

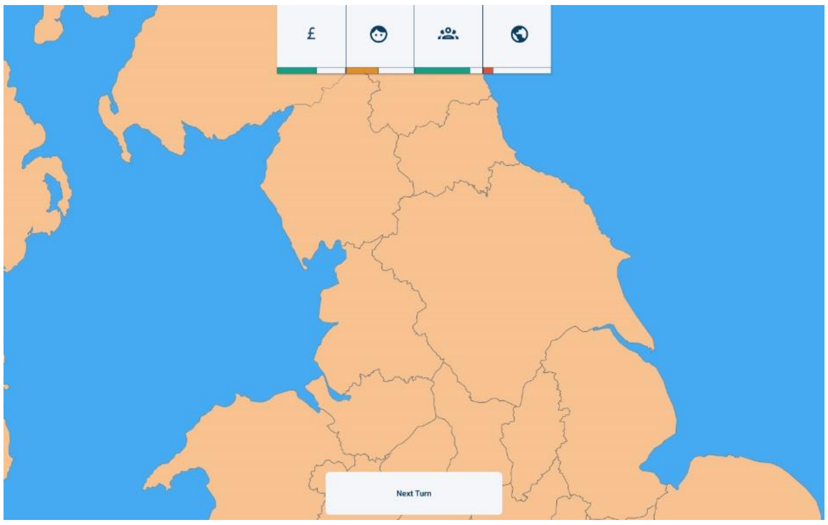
Attribute UI



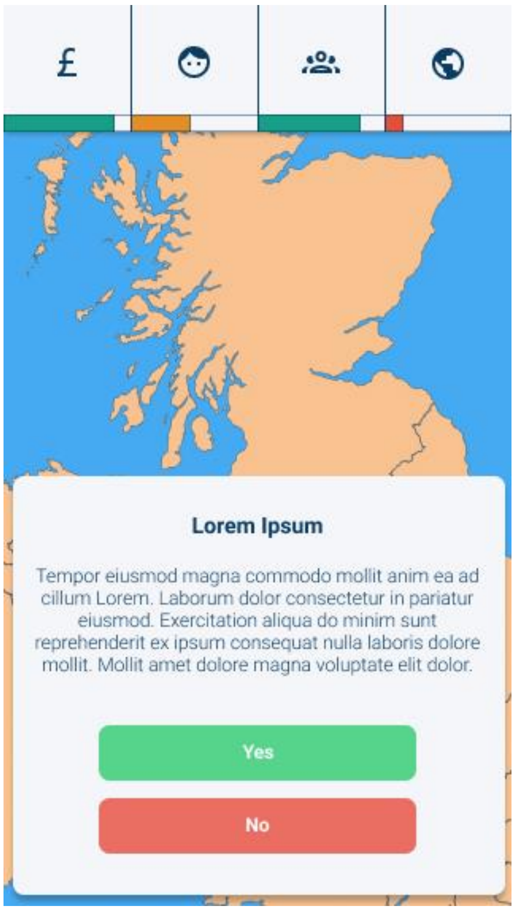


Decision UI Stage 1



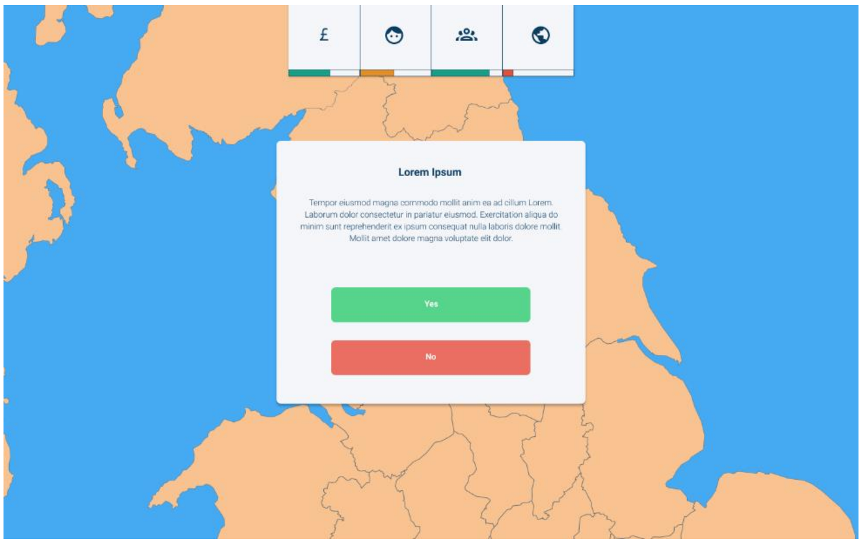


Decision UI Stage 2



Lorem Ipsum

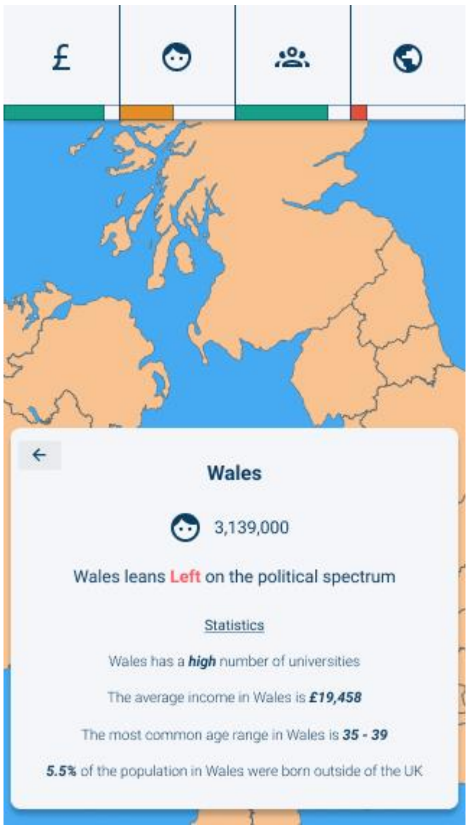
Tempor eiusmod magna commodo mollit anim ea ad cillum Lorem. Laborum dolor consectetur in pariatur eiusmod. Exercitation aliqua do minim sunt reprehenderit ex ipsum consequat nulla laboris dolore mollit. Mollit amet dolore magna voluptate elit dolor.



Lorem Ipsum

Tempor eiusmod magna commodo mollit anim ea ad cillum Lorem. Laborum dolor consectetur in pariatur eiusmod. Exercitation aliqua do minim sunt reprehenderit ex ipsum consequat nulla laboris dolore mollit. Mollit amet dolore magna voluptate elit dolor.

Province UI



Wales

3,139,000

Wales leans Left on the political spectrum

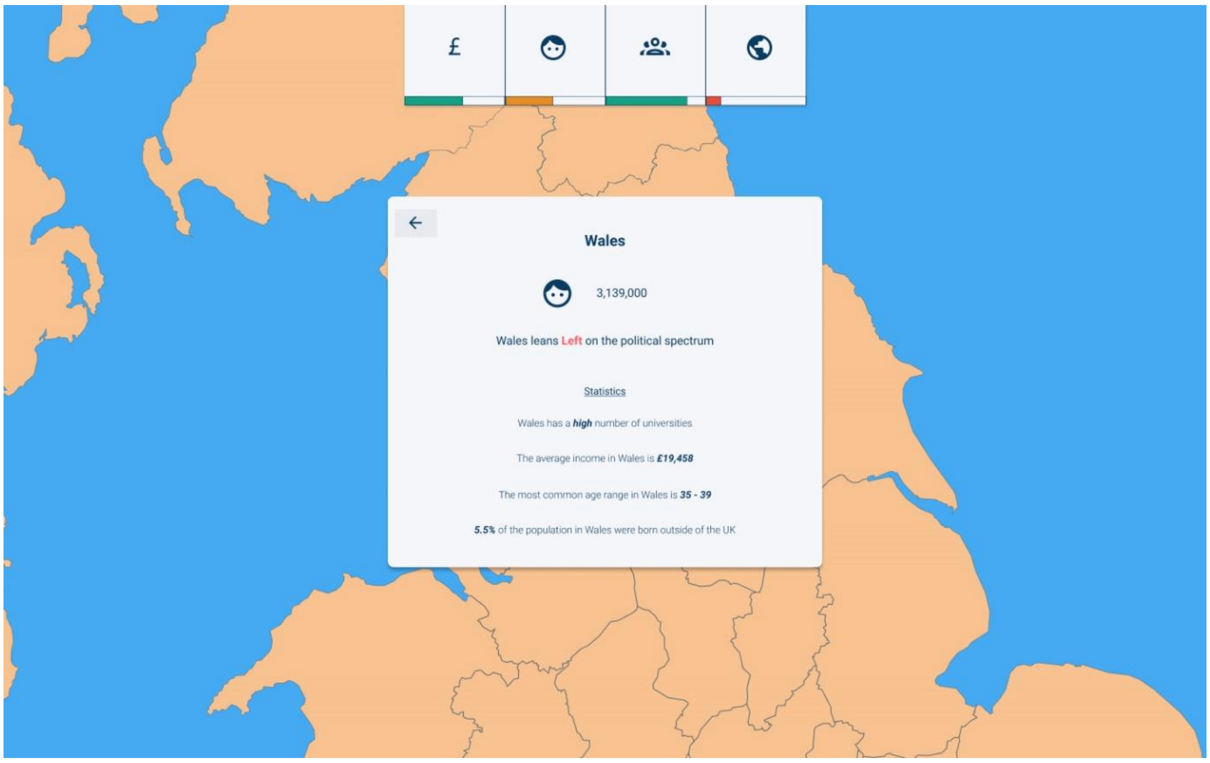
Statistics

Wales has a ***high*** number of universities

The average income in Wales is ***£19,458***

The most common age range in Wales is ***35 - 39***

5.5% of the population in Wales were born outside of the UK



Wales

3,139,000

Wales leans Left on the political spectrum

Statistics

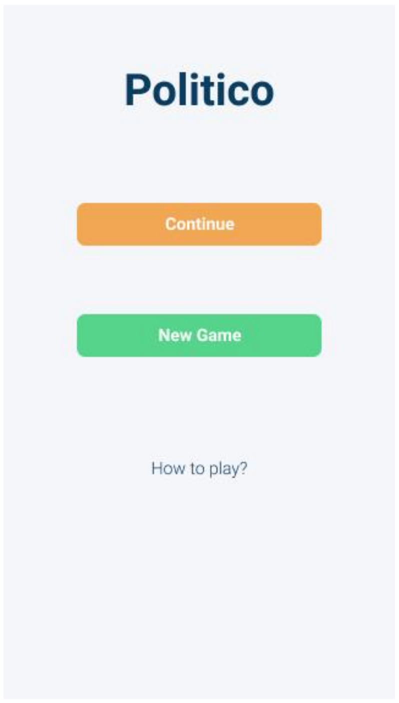
Wales has a ***high*** number of universities

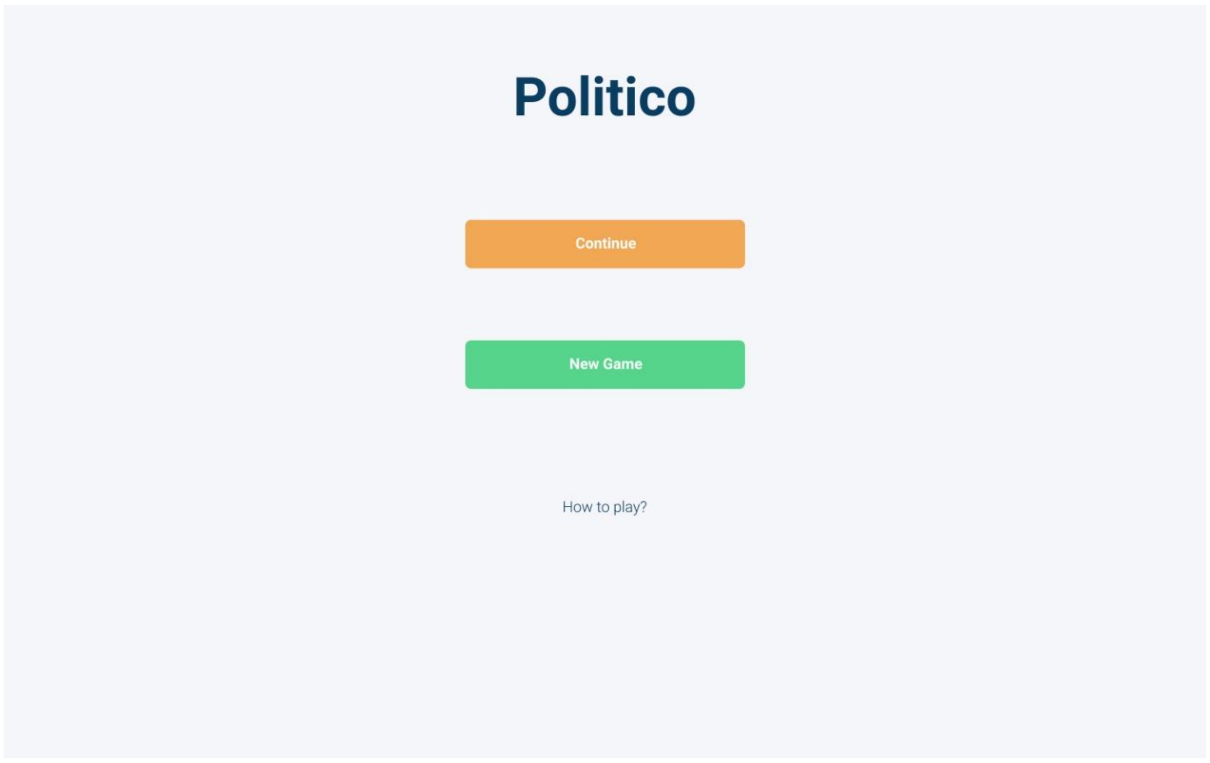
The average income in Wales is ***£19,458***

The most common age range in Wales is ***35 - 39***

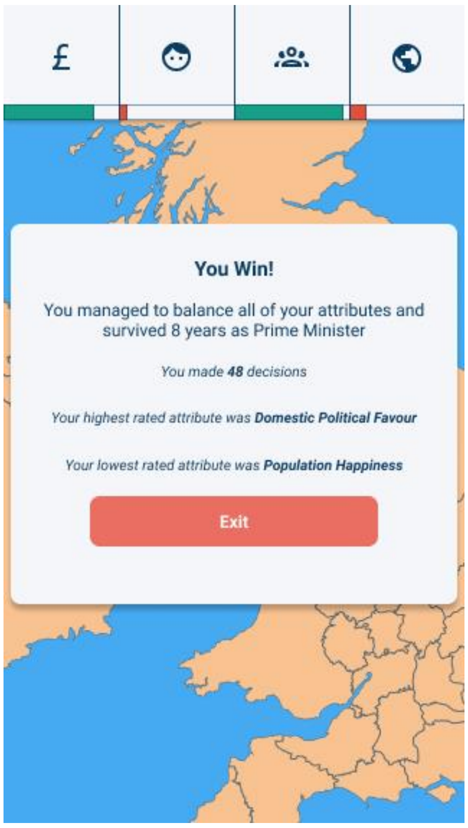
5.5% of the population in Wales were born outside of the UK

Start Screen





End Screen



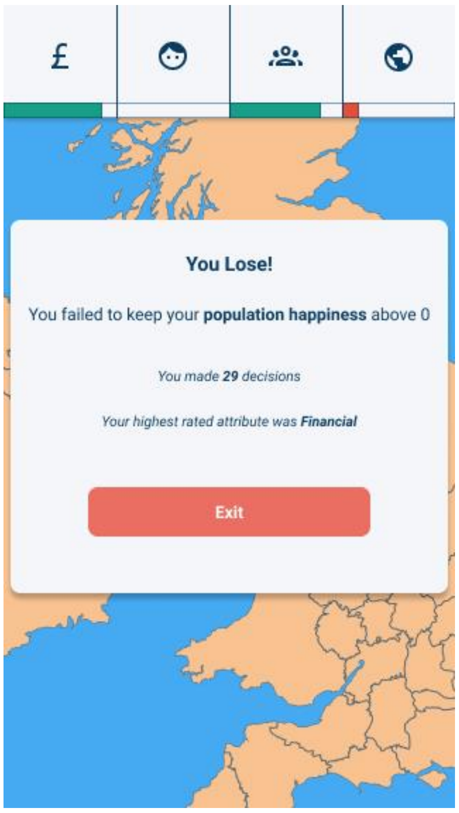
You Win!

You managed to balance all of your attributes and survived 8 years as Prime Minister

*You made **48** decisions*

*Your highest rated attribute was **Domestic Political Favour***

*Your lowest rated attribute was **Population Happiness***



You Lose!

You failed to keep your **population happiness** above 0

*You made **29** decisions*

*Your highest rated attribute was **Financial***