Background

Akkroo's software helps businesses collect customer data at many different kinds of events. To be most effective, the data capture software we build must be flexible enough to suit many different use cases. Data is collected from a consumer in one form or another (form questions, photographs etc), processed and then stored ready to use. Building flexible systems is hard, and so we are looking for developers who are up to this challenge.

Your task

- Create a small data capture system that is flexible enough to be able to create many different **components** from a library of **reusable UI elements**.
- Below you'll find a diagram of a data capture process, based on a real scenario for our customers which you need to model.
- Your system must be flexible enough that brand new components can be generated without having to make a lot of code changes.

What we've provided you with

- An initial Library of Components (below) that is used in our hypothetical processes. New components might be created in the future that we would want to 'drop into' these processes to change or extend them.
- An illustrated scenario that explains how the application should behave when it is being used.
- A written assertion that covers how the application should behave visually for the user.

What you should deliver

- A compact Javascript implementation
- A well-designed solution and responsive UI that works well across a variety of different devices (laptop, phone, tablet, etc)

Things to consider

- You should be prepared to describe your solution in depth in a followup discussion with us detailing your approach and decisions.
- The UI and experience of the customer-facing parts of the application. We want to see a well considered, rational design along with high-quality, semantically considered code.

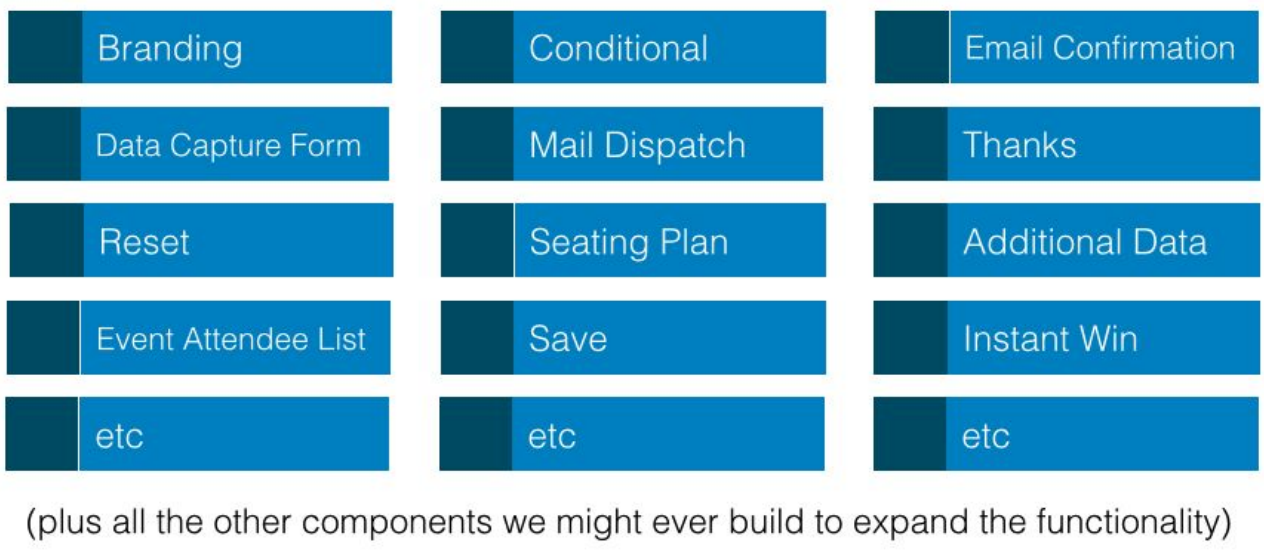
- We will be interested in your choice of tools and your design decisions. Including the use of new or interesting CSS, javascript and design techniques is encouraged where appropriate.

Remember!

At Akkroo we need to be able to design flexible systems. This task is about demonstrating that you can build a flexible system consisting of reusable elements.

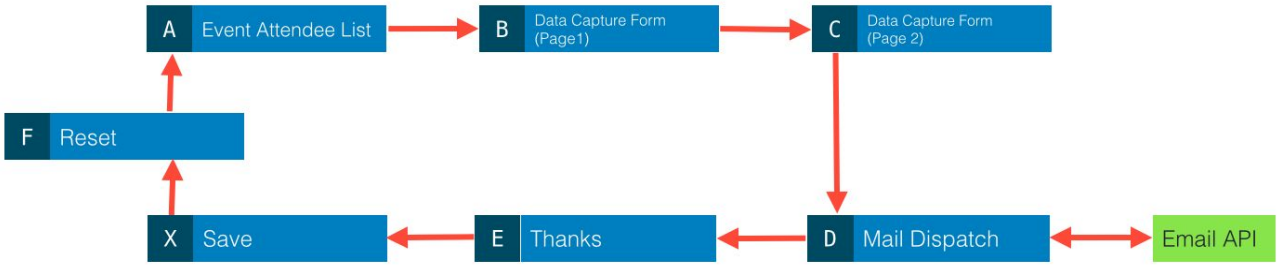
Library of Components

Here is the component library we'd like you to work with (you won't need to build all of these). The processes in the task below are built with components from this library. Review these briefly, then read on!



The Task

Visual description of the Process:



Written description:

John Jameson walks over to the exhibition booth and sees the iPad running the software. He selects his name from the Event Attendee List (component A).

He then enters his email address on the first screen (it wasn't present in his record, so wasn't pre-filled) and opts in the receive marketing emails on the second screen. When leaving the booth he receives a marketing email thanking him for attending.

Next, Sam Stuarts walks over to the exhibition booth. He selects his name from the list of event attendee names, enters his email address (as it wasn't present on the initial record) and doesn't opt-in to receive marketing emails. He then leaves the booth and receives no marketing email.

Later on in the afternoon, Matt Michaels arrives, selects his name on the event attendee list, opts-in to receive emails and corrects his email address which was pre-filled as it was stored on his record, but it contained a mistake. He leaves the booth and immediately receives a marketing email thanking him for attending.

In this process, you will need to include the following components from the library:

Component	Description
Event Attendee List	A list of attendees (from the sample in example_data.json)
Data Capture (page 1)	A simple form that collects the user's email address
Data Capture (page 2)	A simple form that collects a postcode and a checkbox which a user ticks if they consent to receive emails
Mail Dispatch	Sends as email to the user if they have opted-in to receive emails
Thanks	Displays a simple "thanks for signing up" message to the user
Save	`console.log's` the details captured from/about the user (in a real-world this component might persist data to a database) We would expect to see properties like 'email', 'postcode', 'optedin' here
Reset	Resets the application ready for the next user

Important Notes

- You don't need to write code for the API (green box in the diagram), as we have provided a function that mocks this call
- Don't worry if you get stuck implementing this system, you should still return your ideas. The aim of this exercise is to be able to discuss your decision making process, not only assess your coding skill.
- Remember that you can reach out to us if you need more explanation/help on this task!