**Cloud Lab**

**Creator**

**Our proposal is to create a Cloud Lab management software solution.**

**Executive summary**

The idea is to create a GUI desktop or web application that educators can use to build and manage a lab in AWS without a need for writing any code. Amazon does provide a tool called [Cloud Formation Designer](https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/working-with-templates-cfn-designer-overview.html) but is too complex for an average user to use.

What we are looking to create is a drag-and-drop web interface where you create and configure a lab, with the ability to make multiple copies of the lab to accommodate the number of students who will be using it. The goal is to create a very simple and easy to understand interface for rapidly creating cloud labs.

**Requirements**

We will required a server backend to serve responsive HTML pages to the clients, and to call amazons api on the users behalf (Due to security reasons browsers don’t allow webpages to call external API directly, JSONP is a duct tape solution if the API provider supports it, so instead we call the api’s from the server side).

The back end will be designed for a single user, so we must distribute an easy to use server installer.

The client side must be responsive, easy and, quick to use otherwise our use case becomes invalid.

**Outcomes**

To address the usability issues raised by teachers who run IT courses and need to create Labs and rapidly with zero hassle.

These key scenarios would all be solved with an easy simple to use frontend for AWS that provides the users (Teachers) with only the absolute minimum options so that creating a cloud lab or quick server only takes a few clicks.

**Solution**

A simple easy to use drag and drop frontend for AWS EC2.

**Use Cases**

A teacher wants to make a lab where every student has access to 1 Windows server and 2 Linux servers. They use a gallery to select the server types, then right-click to configure each server's settings. After that, they can make 10 copies of this setup, and have all that deployed on AWS.

An It teacher needs to quickly spin up a VM on the fly in a classroom, and doesn't have the time to configure a bunch of settings and keep their class waiting.

A teacher who doesn’t specialise in IT but still needs the ability to create servers for students to use in their studies.

**Team Structure**

Jayden Hawkes

Responsibilities:

Code review

Debugging

Back end API design and part of its implementation

Front end graphics programming w HTML canvas (If needed)

Jayden Aislabie

Responsibilities:

Git technical support

Frontend Developer

Author

Reviewer

Mediator

Backup Secretary (meeting minutes)

Alex Hathaway

Responsibilities:

Developer

Author

Reviewer

Mediator

Facilitator

General knowledge and support

**Risks and issues management**

Django is the back end we chose because it's written in a language everyone understands, However no one in our team has used it. We considered Node.js given one member has experience, but only 2/4 of our members have used Javascript before so we opted not to.

With this approach, We can guarantee that every team member will be able to contribute, the largest risk is using a server backend no one has experience with, however this is mitigated by the fact that it is the most popular python based server backend so it has a lot of community support.

**Development process**

We will follow a fortnightly iterative development structure where the goals of each iteration are discussed as a team and the work is divided up.

**Completion stages:**

**Initial stage.**

UI completion, to the point of creating a amazon EC2 cloud instance in a couple clicks

Server side completion with basic aws api calls, to the point of creating basic EC2 instances

**Further development:**

UI supports creating cloud instances of machines and configuring them with easy drag and drop elements.

Server side allows for creating complex setups as requested by the client and calling the appropriate AWS api calls.