This is the index I was working with:

Index:

The languages that would make sense here would be javascript and mysql, which I have done some on so we should be fine. Only thing I haven't done is work with APIs, so our contingency if we aren't able to sync nicely with icecat, we can 'live webscrape' which I did last semester, or last resort we can webscrape some products and create our own database for the app to refer too.

These were my aims I came up with

Aims and Objectives:

Our aim is to find areas where we can increase Usability, Accessibility and Automaticity in cataloging applications, as well as lure older companies into the digital age of cataloging.

Our objective is to solve issues that current cataloging applications have, and add a solution to solve for the reason why some older establish companies are not making use of digital cataloging applications.

Reason for our aims and objectives is because legacy or new companies who have not yet integrated digitally sit at a disadvantage amongst companies who have, which effects company's success in an unnecessarily negative way.

These were the issues that I found in my research

- Issues companies are facing:
    - Keeping product listings up to date, and extending on the catalogues.
      Time is the main issue here, even with 5 minute for a product listing people still can't find the time to do it, or do it properly.
      The main reason here is most companies don't have a dedicated person who handles this, they delegate it as a task to some employee.
    - Normalizing supplier or third party data.
      Time again here is the issue, having large amounts of data then having a single person decide what data is relevant to present. Or data that is to technical or not technical enough.
    - Managing complex product structures like kits / assemblies.
      This seems to be a problem with sku syncing, which is an issue that would have a few different solution. Boils down to poor product management.
    - Managing products listing across multiple sales channels.
      Syncing product and stock with online shops, catalogues and differently located branches and shops, companies limited on time faces these issues, as well as poor ERP (*Enterprise Resource Planning*) systems.
    - Cost to create maintain and print
      The cost associated with staff or a graphic designer to create a catalogue, publish it online and as well as a print ready version. This is a problem that has been solved with cataloging systems though syncing product and recreating their catalogue on such system brings forth the issue of time.
- Issues customers are facing:
    - Companies hiding the real price of products, concealing additional costs.
      This is an easily solved issue by just being transparent.
    - Complicated layouts, with non-consistency. Causes come from catalogue applications steering away from common practices.
    - Missing product information. Not offering detailed information opens doors to customers not knowing exactly what they purchasing and greater possibilities of returns. This could be caused by the time issue or suppliers not supplying enough information.
    - Too many technical specs
      Customers are different, some don't need to know all the technical specifications while other do. Bombarding a customer with a long list of specs can be overwhelming. This is caused by printed catalogues and catalogues not including 'view more details' tabs.

This is my idea

A web based product catalogue application. A simplistic catalogue creation application that follows the culturally accepted display layout that popular online shops use. The application main feature would include tools to populate their catalogue with rich product information quickly and easily from content suppliers such as **icecat**. The catalogue would be tailored to the company's brand and shared via a url.

This is my aims and objectives for the idea

To create a web based catalogue creator application, which is simple enough for non tech savvy users. The app should tackle some of the issues current applications have, in particular:

- Cost to smaller companies and new users, we want to utilize different income streams, to relieve the burden of cost from the users.
- Ease of product listing and updating, as this is a big issue in cataloging for both digitally involved companies and none digitally involved companies.
- Insufficient and missing product information for customers, we want to create a method in which companies can present all their products details without the time it takes to do so being increased.
- Complicated layouts, we want to steer away from custom layouts that customers aren't familiar with, and promote template designs that are culturally accepted layouts from apps of similar purpose.

This is my scope

Our main focus is to include features that existing applications don't have opposed to including already available features. Within this scope we will briefly indicate how we will deliver each, as well as why we included a deliverable if not yet noted. Within the non-included deliverables, I will state potential beneficial future functionality and their cost implications.

Within this application we will be including:

- User account creation and login system. We will be utilizing a basic database that will hold each user's login data with passwords hashed. As our app security isn't a focus, we will at least protect the user's passwords.
- An interface for the user to create a personalized theme and add products to their catalogue. We will create a theme that is a part of their catalogue that is personalized for each user, as a user logo and colors. As each company would need to be identifiable through their catalogue.
- A categorized index page for the catalogue navigation, automatically created. We will gather the category tag directly from the content supplier. This would help navigating the catalogue for customer users.
- SQL lite databases storing login information as well as data involved in the user's catalogue content. We aren't going to focus to much on database design in this section though will utilize a few tables to achieve our goal.

- API integration with a content supplier, depending on how we deliver the project and if we struggle to fully integrate the API we might need to pre fabricate a database with content. We will create a free subscription to the content supplier's API.
- We will include one or two additional freemium design features that could be included. This will be done to create an income stream for a free application.

We will not be including:

- Multiple content suppliers, we will just be utilizing one content supplier
  - This would be beneficial as some content suppliers will have products that other content suppliers don't have.
  - Cost of implication won't be that great, though will further complicate our application.
- Custom products, that users can include products that are not on icecat.
  - Including a feature that users can include custom products that might not be registered products or aren't on content supplier's APIs is a necessity. Though this is already available on other applications and implementing this doesn't help our objectives.
  - Cost of implication will be great, as this is an entire feature on its own.
- A cart and a method of purchasing through the application
  - Including this feature would be an added benefit to the application, and could be integrated as a freemium feature.
  - Cost implications would be high.
- Site analytics
  - This feature is available in some existing applications, and could benefit company users in determining their catalogues success. Would be a good idea for a freemium feature.
  - Cost implications won't be too high to gather the data, though the cost to implement a dashboard and database would be high.
- Data normalization, we won't be attempting to restructure the content supplied beyond a method that works.
  - This is a necessity for company users, to assure content is all useful.
  - Cost implications would be very high.
- Managing complex products, we won't be attempting to create features for kits and assemblies.
  - This is a necessity for company users, to further customize their products.
  - Cost implications won't be too high here, though too high for this project.
- We won't be integrating the freemium pay methods, just a basic method of including the features
  - This would benefit the application in creating an income stream.
  - Cost implications won't be high, though unnecessary for this project.
- Searchable products and NLP processes
  - This is a feature that would greatly benefit the customer users in navigating the catalogue.
  - Cost implications would be very high here, having a search bar that isn't fully optimal is annoying, and having a search bar that is, is costly.
- Theme customization features

This is a use case uml

Appendix E.1.1