# Considerations and Reflections

**User privacy**

There will not be save any other user data but the email, name and password, and the only thing that will be shown in the applications is the name, I don’t see any big GDPR issues here.

**Restaurant privacy**

The addresses and name will be saved, I’m not concerned around GDPR regarding to this since it is already public knowledge.

**Login & Security**

I’m not strong in this field, but I would like to be. I have worked with oAuth and JWT tokens a little.

When creating an account this application I think its wise to send out an email with a code to the new user, just so they can confirm that they are not a bot.

**Design principles & Scaling**

I’m inspired by the Atomic Design Principle where each component has as low responsibility as possible. When the components are put together, they bring the application alive. In this design principle there a focus on reusable components which I have done with the shared folder.

But this principle is best used for the UI part.  
I don’t think it fits the API, for that I think the thoughts around MVC would fit even though the view is in react.

In most cases design principles is the inspiration, in real world it can be hard to fit all into a box, so I think as log as the design structure makes sense, there is a good understandable flow, good naming and not too many abstractions it will be a good design. Furthermore, I think its very important that any code is easy to debug, the better it is in this area the faster it will be to find bugs and fix them.

I think a bottleneck in this application could be the number of burgers that have been reviewed. To handle this, I think of lazy loading the data table and have a query connection for only reading data and a different connection for writing data.

To help client-side load I think its wise to consider not initialize components before they are needed, even though they are not visible they are pre rendered and ready. This could in a large-scale application use too much memory.