**Pre-Exercise 0**

*Understanding your approach to designing systems*

**Prior Knowledge**

Architecture and design experience

**Objectives**

The objective of this exercise is to understand your approach and level of designing a computer system.

*This is not a test.*

There are no right or wrong answers to this exercise and the results are not handed in. We will have a short discussion about the exercise after.

The aim of this is to lay a foundation for future discussions during the week.

**Software Requirements**

None

**Exercise**

You have been offered the job of CTO of a startup. The startup’s aim is to make it easier for audiences to find live music that they like. To this aim, you need to support the following functions:

* Allowing users to sign up to your website, either directly or via facebook
* Letting bands and performers to sign up and post their gig listings, including dates and locations.
* A system whereby bands can directly link their event listings from Facebook and Songkick (as well as other band listing services) into this site.
* Expose APIs for other websites to post or read listings automatically
* Letting fans follow bands
* Allowing a fan to get notifications of when gigs are coming up within a specified geographic radius of their location
* Linking to existing ticketing systems so that this website can sell tickets directly.
* Linking to a payment system to support accepting payment for tickets

We only have 30 minutes for this exercise, so this is not about designing a real system in half an hour!

Instead, please take the time to think about the core technologies you might use to build this system, the design approach you might take. If you have experience with using SOA, would you use it for this system? If you don’t have experience with SOA, what technologies do you know that would allow you to build this? What are the challenges in building this?

Any work you create for this exercise is just for your own use, but you might consider drawing some quick and dirty versions of: interaction diagrams, system architecture diagrams, models, schemas or other diagrams that might help you.

I hope this “baseline design approach” helps you later in the week as we revisit these types of design decisions.