**Thesis Structure**

1. Abstract
   1. Concise summary of the thesis.
   2. This isn’t a blurb, someone who reads this should have a solid idea of what the paper will discuss.
   3. Clear problem statement; layman’s version of the problem, and why it needs to be solved? Why’s it interesting, what is interesting about it?
2. Introduction **[a subsection of literature review]**
   1. Where is the industry up to at this point.
   2. Introduction should include everything on this topic (problem statements, developments, etc) before I started doing any work; context.
3. Literature Review (a subsection of introduction?)
   1. Where the industry is up to.
   2. Any existing gaps.
   3. Include how to test these systems in the literature review:
      1. What has been done in academia/industry to solve, what are the shortcomings? One shortcoming may be that there is no standardised way to measure the privacy of these systems.
      2. Data hacks and the ramifications.
   4. At the end of the literature review, it should be very clear about what it is I’m going to be doing, this will lead into the methods section; what is the problem that I’m trying to solve.
   5. May have a theory where if I run an index of tests, I have an idea of data privacy/security or something like that.
   6. Literature review gives the thesis a direction; it gives the background for justifying what I’m going to do and how I came up with the thesis. what the thesis will be aiming to prove.
      1. How do we actually test that something is safe and secure? We want to propose a way to do this, the implementation will provide a test bed for this.
4. Methods
   1. Can also discuss meeting with WaiKong here.
   2. Interviews with potential stakeholders, OAIC, etc.
      1. User requirements from these stakeholders.
   3. Multiple system designs.
      1. One central database (all eggs in one basket).
      2. Split databases (split the risk).
      3. The trade-offs of each system
   4. Design tests before the system; if you build the system first and then test it, you know what works and what doesn’t. What you want to do is break the system first, and then build the system that’s unbreakable.
   5. So testing a couple things:
      1. That it behaves as it should given a compliant user.
      2. It behaves as it should given an incompetent (find a better word) user.
         1. System should give reasonable results, or lock the user out.
      3. It behaves as it should for a malicious user (hacker)
         1. Be aware of hacks for this kind of system.
            1. Buffer overflow.
            2. Etc.
            3. Talk about any hacks as well!
   6. Also talk about the rationale behind my system. Solutions exist, why’s this any different?
      1. Free to use? Other options may require a license.
      2. Doesn’t serve the needs of the user; too generic or too specific. Scalable system.
   7. ER diagrams, data-flow diagrams.
   8. Test plan; how the design will be tested.
   9. Design a system where I’m going to compare what happens if I keep everything in one database, or if I separate the databases in one way, or separate the databases in another way; 3 cases.
      1. Keeping all the information in one database is already the worst-case scenario; this is the control condition.
         1. Vary in 2 other ways to see which way is better. Hopefully these 2 other ways are better than keeping all the information in one DB.
         2. If the other 2 ways don’t then this contradicts the theory of separating the data.
            1. The theory is that if pieces of information are broken up in different ways, it should be safer.
            2. Test this with different test plans.
   10. User requirements 🡪 design (3 systems) 🡪 test plan (tests which of these systems is best; include a hypothesis) 🡪 preliminary work that has been completed (e.g. “I’ve started using matlab with sqlite and created the control condition where all info is kept in 1 database; here’s the code for my prelim work) 🡪 plan for thesis B (write the other 2 databases in code, run them with the test plan) [this is what I’m going to do with the 15 weeks of the second semester; have 3 different ways of running the databases and I’m going to test them to make a recommendation]. Also include some suggestions: “if I had more time, I’d make these changes due to flaws XYZ that I found”, suggested remedies. Due to time constraints, this is left for the next student 😊.
5. Results
   1. Self-explanatory.
   2. Which system is best and why?
   3. Any unexpected results?
   4. Any road bumps along the way?
   5. Comparisons with existing solutions.
6. Conclusion
   1. Final judgement and suggestions with every other piece of research in mind.