SUPERVISOR MEETING MINUTES

Date:	20/10/2015
Time:	1000hrs
Venue:	Level 5 sofa outside Prof Chris' ofice
Attendees:	Claudia Foong Pui Shuen , Goh Yi Xuan, Karen Lim Wen Yan, Nguyen Luong Thanh, Vu Hoang Minh
Agenda:	1. Update him about what our project is 2. Show him the architecture and highlight the fact that most of our data are from web services 3. Backend architecture - node.js vs java 4. Advice on project acceptance preparation 5. Expectations for project management 6. Considerations to think about 7. Other advice

No.	Task	Follow Up(Person-In-Charge)	Deadline
1.	**Update him about what our project is:** ➢ Web Application that visualises data for logistics SMEs ➢ Helps them to easily understand and track their business performance through their KPIs (costs, revenue, delivery fulfilment rate, order lead time etc.) ➢ Our application does API calls to extract these data from our sponsor's application, VersaFleet and its partnering softwares such as Xero ➢ Not a fleet management systems	-	-
3.	**Backend architecture - Node.js vs java** **Node.js** ➢ Facilitates integration with frontend as we are using react.js for frontend ➢ Facilitates code push to git as there is no need for compilation ➢ Changes to codes can be seen without the need to recompile	-	-
4.	Advice on project acceptance preparation: ➢ Be knowledgeable of our application - Why we chose certain technologies over		

	others? e.g. React.js: Sypher Labs is currently using react.js framework for their frontend development → Easier for them to maintain when we handover the project to them in future ➢ Present the % of completion for our project → Shows that we are clear of what we are doing ➢ Codeship automates testing and deployment → With a small pool of employees, Sypher Labs can focus on fixing bugs and updating features rather than conduct tests ➢ Get client's approval for paper prototype by acceptance → Shows that our application is developed according to client's request		
5.	**Expectations for project management** **Planning every other thing that isn't dependent on the coders:** ➢ Number of user tests and when to conduct these tests ➢ Have an agenda for client meetings ➢ Have progress updates for every supervisor meeting ➢ Have a realistic plan which includes proper risk mitigation - If one person is unwell, someone else must be able to cover his work. So there must be at least two persons capable of doing the same work.	-	-
6.	**Considerations to think about:** ➢ A successful application should do the work rather than create more work for its users ➢ The application we build MUST actually help users solve problems/reduce work/save time etc. ➢ Does the adoption of our application increase workload for operations manager? ➢ How do we know if we have failed if clients are not present during the final presentation? ➢ **Are we worth the changes required?** ➢ Some changes could cost employees their	-	-

	jobs and hence the resistant it the adoption of our application
	➢ Work closely with clients so that our product will be adopted by them
	➢ Have data to prove that our application is feasible and useful
	➢ Have a backup plan if clients we are serving change - clients quit etc.
	➢ How to measure success? or failure
	➢ What data can we show? Time it took to complete certain tasks?
	➢ Provide the option to sort:
	   ✓ Top 5 important information to display to the users
	   ✓ Have an algorithm for this
	➢ **Think through the business**
	➢ Which are the most important graphs out of those we've build
	➢ How did we prioritise?
	   ✓ User study shows that they kept going to the same graphs/tables
	   ✓ Why are the data organised in a certain manner?
	   ✓ It would be good if we can do "machine learning" → preset the app and show graphs that users would usually view to them
	➢ Usage of technology must be explained with what our focus is		

7.	**Other advice:** ➢ Start with features that are more daunting so that we can mitigate any risks early → convince client that the feature is not very useful through our research or drop the functionality etc. ➢ Before building the application/changing features/pages etc., build a paper prototype and show it to our client → to identify what they like/dislike about the application ➢ Have clients "sign"/approve the paper prototype before developing them to avoid development complications when major changes are required ➢ People often do not know what they	-	-

| | | want/don't want until they are shown the application
➢ Acknowledge that we are limited by Versafleet users. Although the number of people we can help is small, we MUST prove that the application is indeed useful to them. It can potentially help more companies. Be the reason they want to use Versafleet as well. As long as it helps
➢ Eventually if the client does not use our application, there has to be a lesson learnt e.g. we discovered that the client doesn't like xyz, we helped the client confirm that an off-the-shelf-solution is better etc.
➢ What makes the client great? (X Factor)
   ✓ Helping "dying" traditional logistics SMEs
   ✓ Is he targeting a niche? | | |

Vetted by: Eva

Follow up: To be circulated amongst team