**HW Week 8-Ind: PRD Examples**

Name: Aarjav Dutt

Part 2. Read the “How to Write Good PRD” paper. Describe:

* Ten steps to a good PRD
* 12 common pitfalls

Ans. The 10 steps to a good PRD are:

* + 1. Do the Homework: This step includes studying the customers, the competitors, your own team’s capabilities and the available technologies
    2. Define the product’s purpose: The PM should have a very clear proposition about the product that can be easily communicated to everyone from his team, to the customers
    3. Define User Profile, Goals and Tasks: In this step the PM works with the Product designer to classify the types of customers and users, their goals and objectives, and the tasks that will help accomplish these goals.
    4. Define your Product Principles: Product objectives and principles must be clearly identified and defined, so as to lead the product team in one direction and avoid clashes between teams later.
    5. Prototype and test the Product Concept: The most important step is to create various forms of prototype for product validation. These prototypes will be tested and validated under different kinds of testing, before the product can be built.
    6. Identify and Question your assumption: Once the problem is identified, the assumptions made earlier can be identified and questioned.
    7. Write It Down: PRD works as a conversation between the entire product team and hence it must be documented and updated throughout the project.
    8. Prioritize: After getting all the requirements, they must be ranked and prioritized so that any important feature is not left before the product is shipped.
    9. Test Completeness: After the PRD is fully written, It must be tested if it has all the information needed by all the stakeholders and adding and details or any clarifications must be made
    10. Managing the Product: During implementation of product, If the requirement or any specification changes, it must also be updated in the PRD as well.

And the 12 common Pitfalls are:

* + - 1. Usability Testing Too Little Too Late: Usability testing tells the product team how bad the customers struggle with the product, but also quickly tells them what needs to be done to correct it.
      2. You are *NOT* your customer: companies should not assume that if they like the product, their customers will like it too. To expand the customer base, reality checks must be done with the product to understand the target market.
      3. What versus How: The requirements should clearly state the problem that is to be solved and should not contain the solution itself. Giving the solution in the requirements can be dangerous to both the Product manager as well as the development team. It shuts doors to better solutions for the same problem.
      4. Too Little Detail: If requirements are under specified, the development team may assume something is not required while it actually is, and the missed-out details can lead to confusion between the stakeholders.
      5. Too much Detail: Giving out too much detail to develop can become burdensome, as the time in which it gets complete, that it may become obsolete.
      6. Less is More: Adding too many features causes the product to be more complex and needs more customer support and related costs. The main product should be kept in focus and features may be added as the product evolves with time.
      7. Engineering-Driven Requirements: The PM should be aware of not to get overwhelmed with the engineering organization and have requirements that will cater to them, as it changes the focus of the product and it no longer is useful to the customer. The product must be the one that can be executed with strong effort and not driven solely by the engineering.
      8. Customer-Driven Requirements: The reverse of above situation is when sales or marketing influences the requirements of the product. These requirements are influenced by special customers, and thus neglects other customers and infeasible for the company to develop custom solutions for each customer.
      9. Emotion can be Logical: Prioritizing the requirements can be a challenging job for the PM. This is where the emotion plays vital role. With a deep understanding of the customer, requirements are decided that can elicit an emotional response.
      10. Standard is better than Better: Just for the sake of innovation and providing something different, the existing standards must not be changed, and the product should focus on what the people actually need.
      11. Pay your Taxes: The engineering team after the release want re-architect the product to improve scale and security. These re-architectures take longer time than expected and cause losses to company. A better solution is to give some percentage to engineering so they can scale and use resources every year and it doesn’t cause trouble to the company.
      12. Requirement versus Design: Creating a better design or user interface is equally important and should not be ignored while creating requirement specification. This is because adding a design later, after the implementation has started will cause major perturbations to the engineering team. Hence design must be completed before the implementation of requirement begins.

Part 3. Read the three PRD examples. Which example is your favorite and why (in terms of following the template and taking into account the takeaways from the paper above)? Write a short paragraph, defending your answer.

Ans. Of all the 3 examples of PRD given, I would again go with the idea I selected among the 3 best ideas assignment, i.e. Barco or the boat rental app. I think this is the most appropriate example of a perfect PRD. It covers all the topics of the PRD template as well as follows the ten steps for making a good PRD. It is detailed, well explained and contains graphics for the user’s better understanding of the product, which should be the primary goal, when drafting a good PRD. Also, among other things, I observed that the other 2 PRD had a few sections missing or not very well explained. For example, for MBACAS, the walkthroughs were pointed out in bullets, which is not very useful as compared to Visual or Verbal wireframes given in other 2 examples. Similarly, in the Sprout app, The Detailed Design & Features Description section felt missing of major information like MVP, vNext, vLongterm. The Features Description section should be the major part of the whole document, which I felt had only been concise in 2 pages. Thus, keeping in mind all the mistakes in the other 2 PRDs and the merits of Barco, keeping it very articulated and visually helpful, was my favorite PRD.