Bo Peng

1．

Graduation is coming soon, and peer pressure from people like Mark Zuckerberg who started Facebook and brought revolution. Existing solutions are not satisfying, which different systems don't talk to each other, or update in real time.

2.

User use the calendar without paying anything, but they do contribute their browser usage and user data. Advertisers pay for ads for the calendar. Their initial choice of launch is university. They are confident providing cheap, high quality calendar service will interest more people.

3.

Web App based by databases provides better user experience. Simple sign-ups. Social network data may be useful. New advertising - drag and drop.

4.

They brainstorm ideas, offering suggestions and eventually sell the Calendar. They start next venture after this.

5.

Yes, they anticipated that. I think their decision to sell Kiko for their next venture is wise and reasonable. They could hardly compete with Google Calendar with Gmail.

6.

Their goal was the build of the world's first AJAX calendar APP. There are millions of potential users and they want to sell the site to Yahoo, Microsoft or Google eventually. They believe AJAX calendar like Kiko has an inherently sticky user experience and they have first-mover advantage. I think "Cute hacks can cost you time" could be deduced from the case study information. It's not that important that everyone could have the same problem.

7.

Hire Slow, Fire Fast; Get investors involved and Build incrementally would be three most likely valuable aspects as to me. Because programmers are the base of your product and code. If you can't find programmers who are good at their jobs, you could hardly make your product developed as you expected. If investors could involve in the build of the project, you will know better what should include in your features and what should not. They usually have a better horizon than programmers on what potential customers would like to have. What's more, build code little by little is similar to what we learned in class: Agile programming. Developers could easily get bugs fixed and features changed according to customers need. Developers would have a lower chance of doing non-sense.