**"How to choose the platform on which to build your bot?"**

Think about your particular use case and the demographics you are trying to reach. For example if you wanted to reach [40% of teenagers in the United States](http://www.prnewswire.com/news-releases/kik-receives-50-million-investment-from-tencent-300130265.html) you would build on Kik. If you wanted to solve an enterprise problem for [3 million daily active users](http://venturebeat.com/2016/05/25/slack-passes-3-million-daily-active-users-930k-paid-seats/), you would build it for Slack.

Think widely about platforms & bots. Poncho, the weather cat, was originally on Slack before it was on Facebook Messenger but it happens to be more popular on Messenger.

**"There are so many bot frameworks around. How do you decide to pick one?"**

Consider flexibility & what happens if you get stuck on one platform. Do you want to build only for one platform? Would you want to scale to different platforms? Does the framework provide integrations and support interoperability? You can use some frameworks without writing code but they can be hard to plug in to different platforms. Some other frameworks provide integrations for different platforms but require considerable technical knowledge. Exploration necessary.

**"How do you give your bot personality?"**

*"Drive down to LA and hire a writer!"* Jokes apart: Find someone who isn't directly involved with making the bot and talk to them as the bot. Make a note every time the conversation becomes awkward and note how you navigate during such situations. Note what happens when people deviate from the flow and also what makes them do this.

**"Don't lean on ML as a crutch."**

Don't try to do and be everything. It can be interesting to try and handle everything that a user can throw at your bot. This can work if you get it right but very risky if you don't. Also think about the platform and how people use it. For Slack, you don't need to worry about "witty repartee" because people need to use it for work. Think about interactions and how the bot is intended to be used.

**Get feedback at each step.**

If a step requires a user to provide some input or information ask for verification. If there is an important step, ask them to confirm. Don't assume.

**"Set expectations"**

It is important to keep people between the scope & context.

**"Don't try to be clever"**: People know not to expect much from a bot so don't create false expectations.

Quick example: "hi i am a bot, i am pretty stupid but i am here to help you"

If you set expectations low, you will always exceed them.

**Bot conversations are not stateless.**

You need to keep state and track them. This can be multiple conversations and multiple channels.

**Web/mobile fall under the CRUD model but Bots are state machines.**

Each conversation is a state in a state diagram.

Conversational User Experience is basically designing & building a non deterministic state machine. Think of a potential conversation in terms of states & figure out how you can reach that state.