CIS4301 Guest Speaker: App Prototyping

Speaker: Dave Stanton (@gotoplanb)
Notes: Ryan Roden-Corrent

February 19, 2014

1 App Development

1.1 Users Goals Metrics

- Who is the user? What are their goals?
- Come up with a "persona" to represent the people who will be using your app.
- Have general "metrics of success"
- One-size-fits-all software doesn't really appeal to anyone in particular

1.1.1 example

Developer

IT Manager

CIO

1.1.2 5 person useability studies

Often just testing with 5 people can give great insight into what works and what doesn't in terms of the user interface. Track the errors they make (thinking they were clicking on something when they really weren't) and tweak the software. Need to make sure you have the appropriate testers for the software.

1.2 Strategy

Exit Strategy What will you do if it takes off? Is it a full time project or just a project done out of curiosity?

Defense What makes your app different from what already exists?

Funding source dictates how you can build features

1.2.1 Funding

bootstrap Your own money, you dicate how to build the software

grant grants tend not to care as much how it is build, just whether it achieves the end goal

With grants, need to know when to stop. Don't undervalue your time.

1.3 Flow

Flow Diagram Create before writing code, can save time in the long run.

Wire Frame Representation of user interfaces. Each wireframe should have an associated persona and goal. Appearance not important, very high level

Composite Mock-up, generally looks better than wireframe

Paper Prototype Walk someone through using a paper version of the app

Define MVP Minimally Viable Product: what is the minimum required for app to "work"

Prioritizing Features what are the most important features? What are blocking features? In what order should features be completed so everyone can work in parallel?

Tickets use goal tracker (e.g. Trello) and issue tracker (e.g. github issues.)

Sprints define a time period for work, set backlog during planning meeting, typically have once-a-day $\approx$ 15min "standup" meeting

Quality Assurance mutliple environments (work, staging/integration, production). Test in an environment exactly like the "real-world". Run test suites.

Release

Retrospective

1.4 Common Trouble

Scope creep Keep adding new features, which delays release and bloats product

Nebulous Hierarchy can't have a "flat" team. Assign specialties or lead roles to members.

Too many stakeholders Who "says yes" for a particular kind of decision?

Automating too soon don't over-engineer too soon. Just make it happen as quick as possible

1.5 Stack

Make sure you are using the appropriate technology for the project