# Work Smarter, not Harder

Start building things you know are Good Ideas™

James Hou

# Overview

1. About me
2. Soft skills I use on the daily
3. Anatomy of a project
4. Slice of my week during a project
5. So what?
6. Extra Credit: How to lead meetings

**These slides can be grabbed here**

# About Me

Psychology major switched to B.A. in International Business.

Self-taught - hobbyist techie to **problem solving** professional.

**9 years** in ecosystem. Accidental Architect @ Colliers => Independent Consultant @ Google.
Startups, big firm consultant, freelance, digital nomad, **tiny** & **large** enterprise teams.

Embraces both **business** and **technical** aspects of the platform.

20% Business Analyst / Scrum Master / Product Owner.
80% Architect / Developer.

# Soft skills I use on the daily

# Beginner

## Active Translations

Being able to **turn techno-babble into layman's terms** while still making your point or collecting information.

Business stakeholders will like working with you (that's pretty important).

## Be a listener

Everyone coming to you with a problem is trying to solve something.

**Listen**, **acknowledge**, and even if you disagree, you're **looking for what they need** (even if they don't know yet).

## Be transparent

**Voice** concerns, stress levels, what you can and can't do within your skills and knowledge.

Only two outcomes come from this. Your team hears you and **supports you** OR your team hears you and they **don't support you**.

# Beginner+

## Document your bandwidth

If you don't have a backlog, start it now.

**Visual representation** (spreadsheets, sticky notes, JIRA etc) of workload helps stakeholders understand bandwidth, and thus priority.

Is your work item **more important than** what you're currently working on? Ask that, but **with tact**.

## Turn taking

During discussions, take the lead and **have everyone add their thoughts** to the current topic.

**Be open**. Even if it sounds like a bad idea, the new perspective might allow for a mix of ideas or **new perspectives** altogether.

Facilitating this makes people naturally inclined to **include you** in conversations because you are a "**harmonizer**".

# Medium

## Empathy

We're all human. And busy. And not everyone is an expert at building solutions on Salesforce.

Be **patient**, understand where your users/stakeholders are coming from and allow them **their point of view**. Even if you disagree, **don't cut them off**.

## Tact

There are no stupid questions. Maybe some from ignorance. Maybe some from misunderstanding.

**Disagree with tact**. Calmly explain why something might be a Bad Idea™.

## Humility

We all make mistakes. I'm sure you do too. I sure do.

**Admit it, then fix it**. Or, correct your knowledge. Hubris makes you really hard to work with. Don't let pride get in your way.

Medium+

## Keep conversations focused

Business is complicated. Real life doesn't work like in Trailhead modules (sadly).

When topics begin to go on tangent, **gentle reminders** about what the **current goal / action item** at hand goes a long way to help sync everyone partaking

## See the xy problem

We all get caught up in solutioning. Sometimes we have to ask ourselves:

**"What is the original problem we're trying to solve"?**

This is related to keeping focus, but this also helps pivot the conversation on rallying around the **original problem**, not the **attempted solution**.

# Advanced

## Solution-on-the-fly

This is a blend of **active translation** with coming up with **candidate** solutions while in any discussion (whiteboard, meeting, walking to lunch etc).

Being able to effectively communicate

## Negotiation

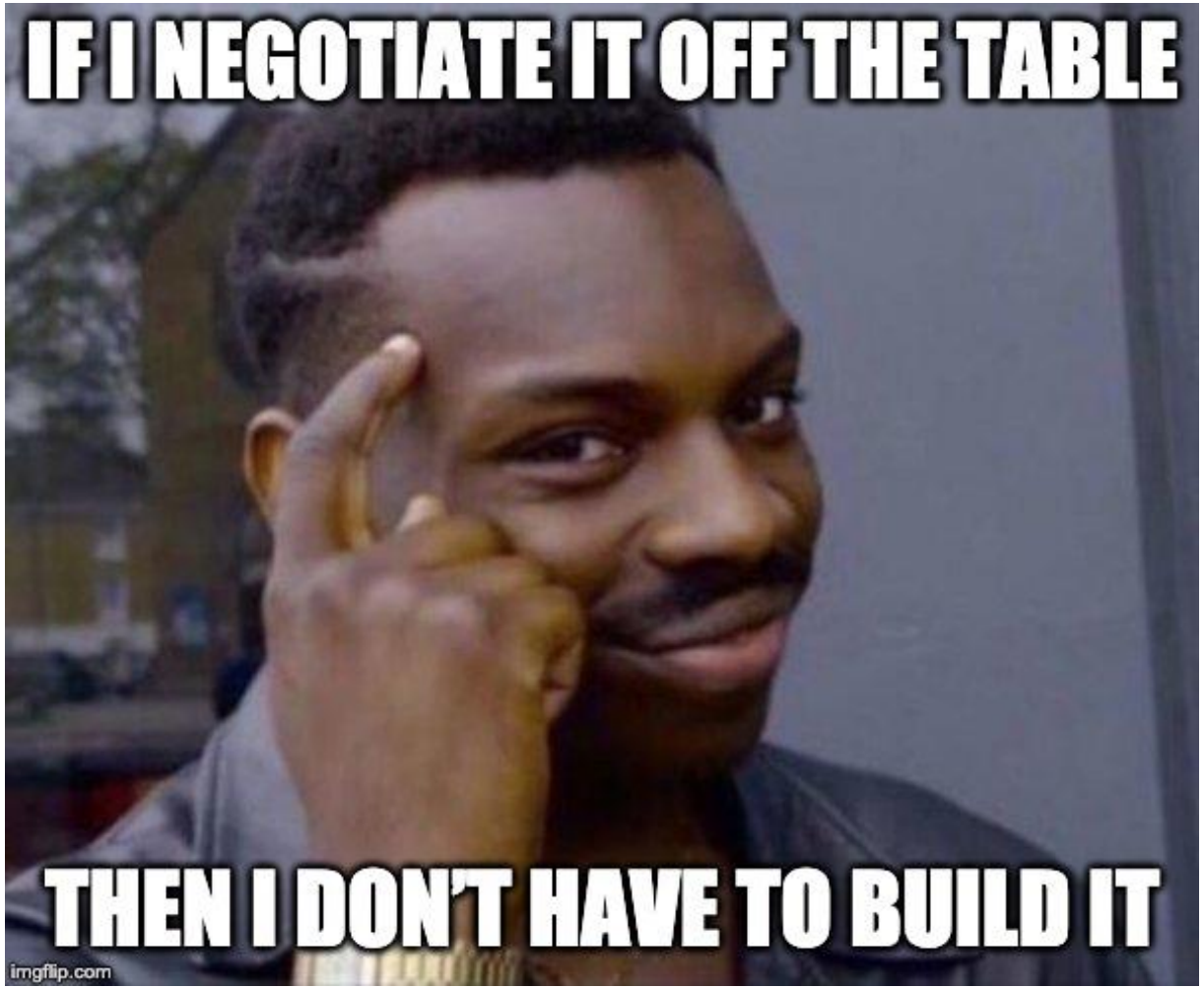
Don't want to work on stuff that is destined for failure?

Do something about it! Head it off by **negotiating it off the table** before it becomes a trainwreck (more on negotiation later).

## Establishing trade-off

However, sometimes it just has to be done.

In these situations, being able to weigh the lesser of two evils and communicating the **LOE, ROI and risks in layman's terms** will steer the project towards success


IF I NEGOTIATE IT OFF THE TABLE
THEN I DON'T HAVE TO BUILD IT
imgflip.com

# Bonus: The User Experience

## Pick a design ethos

Not a soft skill, per se, but having a well grounded **team-oriented** approach to solution design is going to be key.

**Consistency is key**. As more modules are built out, keeping the same user experience consistent makes learning every subsequent module easier.

For in-house admins & devs, **SLDS** is a great design pattern to align to.

## Taking their perspective

Clicks matter! Positioning matters!

Take a brand **new user,** no knowledge of the system. **Could they navigate your solution**? Could they navigate your ecosystem?

## KISS (for the user)

Err on the side of less complexity on the UI.

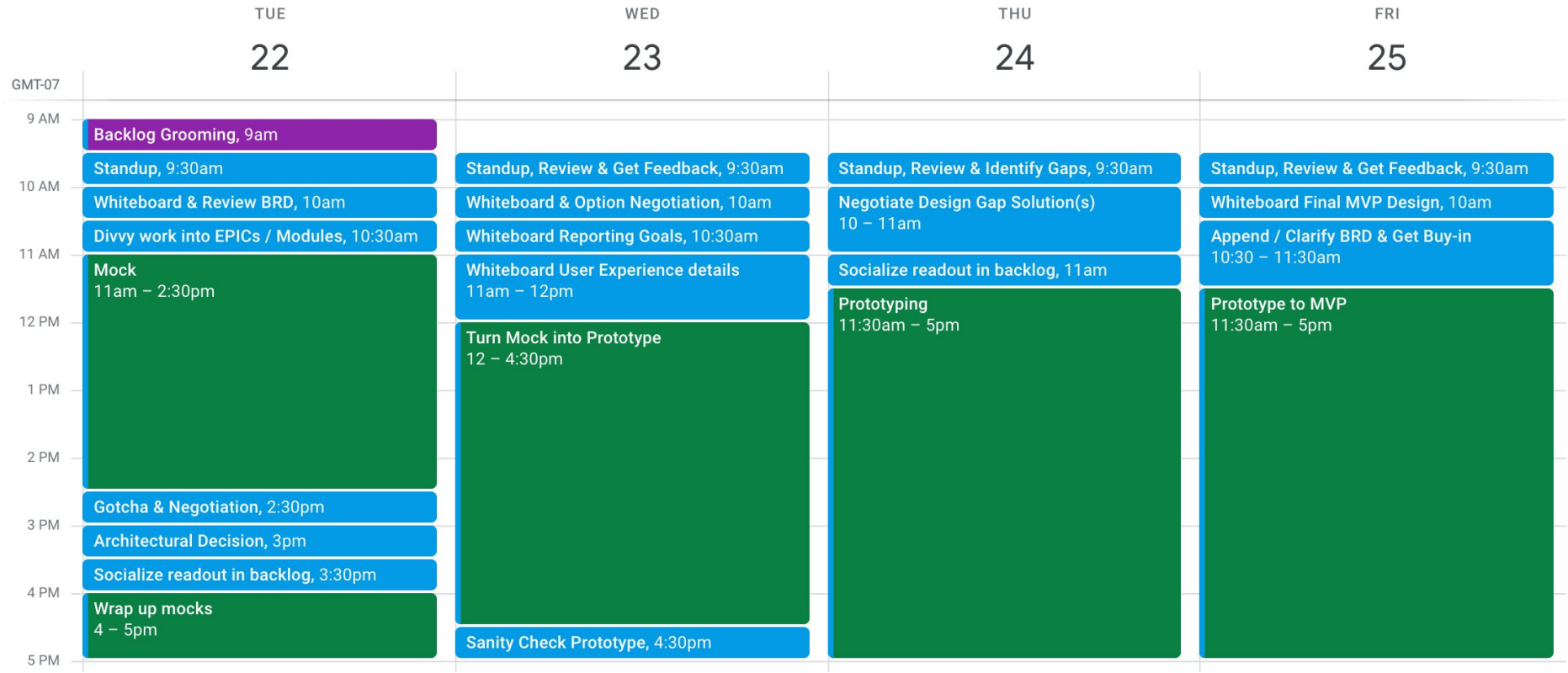
If it means a more code due to less user inputs or to save clicks - make a trade off between UX and complexity/maintainability.

# Anatomy of a project

# Project flow

1. Business Ideation / Strategy
2. **Codify requirements into a document**
3. **Negotiation** / priority on initial requirements
4. Chunk requirements into modules
5. Create **wireframes** and **prototypes** on modules
6. **Iterate and negotiate** on feedback
7. Build modules to MVP (**breadth and then depth**)
8. **Iterate and negotiate** on feedback
9. UAT and **final feedback**

# Slice of my week during a project


TUE
22
WED
23
THU
24
FRI
25
GMT-07
9 AM
10 AM
11 AM
12 PM
1 PM
2 PM
3 PM
4 PM
5 PM
Backlog Grooming, 9am
Standup, 9:30am
Whiteboard & Review BRD, 10am
Divvy work into EPICs / Modules, 10:30am
Mock
11am – 2:30pm
Gotcha & Negotiation, 2:30pm
Architectural Decision, 3pm
Socialize readout in backlog, 3:30pm
Wrap up mocks
4 – 5pm
Standup, Review & Get Feedback, 9:30am
Whiteboard & Option Negotiation, 10am
Whiteboard Reporting Goals, 10:30am
Whiteboard User Experience details
11am – 12pm
Turn Mock into Prototype
12 – 4:30pm
Sanity Check Prototype, 4:30pm
Standup, Review & Identify Gaps, 9:30am
Negotiate Design Gap Solution(s)
10 – 11am
Socialize readout in backlog, 11am
Prototyping
11:30am – 5pm
Standup, Review & Get Feedback, 9:30am
Whiteboard Final MVP Design, 10am
Append / Clarify BRD & Get Buy-in
10:30 – 11:30am
Prototype to MVP
11:30am – 5pm

# So what?

# Consider the following “imaginary” scenario

## Complicated requirements

This is the **first you’re hearing about this**. Looks like it will take **a lot of work** (trust your gut).

If only you had been **involved** in the discussions you could have **steered** this ship away from the iceberg.

## Astronomical LOE

Yep, **after sizing** and discovery, it’s **full custom**.

Can’t even use base components because the design doesn’t align to anything in SLDS.

It doesn’t look and feel like Salesforce (maybe that’s the point though!)

## Complicated solution

Tight delivery timelines, complicated requirements… It’s **spaghetti all the way** down.

Sure it (mostly) works, but why is it *so much* code?

The more code the more points of failure.


OH, IT'S FRAGILE
AND HARD TO MAINTAIN?
BUT EVERYTHING IS TO SPEC
imgflip.com

# Now, the following “imaginary” scenario

## Temper ambitions with Salesforce Feasibility

You were involved in business ideation or a draft BRD was socialized early enough and you saw it.

Scanning the BRD, you **tactfully** present your points on where certain asks have **high tradeoffs** (e.g. high LOE, low ROI).

Remembering the **xy problem**, certain asks just don’t align to it - so they get deprioritized.

## Priority? Order? No way!

Draft BRD is now more realistic in both goals and priorities.

A second, maybe third round, gathering more detailed **tradeoffs** is now socialized.

A chunk of the modules may **align to Salesforce OOTB** (flexipages instead of custom SPAs, SLDS instead of custom CSS).

## Good thing we built it lightweight

Because business, sometimes what you build doesn’t get used the way you think it does.

It’s nice that **OOTB** tools helped bridge some gaps and now the business is more sure about what should happen **post-mvp**.

Good thing you didn’t **invest** all that effort in **full custom** code!

# How to lead meetings

# Standups are for

## Pulse checks

**Raise concerns** to fellow teammates / product owners / SMEs.

If there are no updates, that itself is an update (i.e. all is well).

## Identifying and Removing blockers

Applying the correct resources to keep the sprint velocity up is always key.

Chop down those problems **one step at a time**.

## Defining Daily Tasks

Just so your team knows **which part(s) of the project** is being worked on **today**.

This can head off collaboration issues or get your team to leave you alone if it's a complicated module.

# Meetings are for

## Achieving a goal

**What do you want to get out of this meeting?**

Simple. Powerful. Keep on topic. Topics during meetings should **always** answer that question.

## Strategizing, Brainstorming, Readouts etc

Swarming problems, ideating, and readouts are all goal oriented to collect or pass information to a group of people.

**No Agenda/Goal? No meeting.**

## Defining Action Items

There's always things TODO, even if it's not you.

Identify it, **document** it, **assign** it and **follow up** on it.

# Wrap-up


**These slides can be grabbed here**

http://join.sfxd.org