Minimum Viable Products and Prototyping

Table of contents

Outline . . . 1
Minimum viable products . . . 1
Prototyping . . . 2
Have code generation tools killed wireframing? . . . 2
Key takeaways . . . 2
Next week . . . 2

Week 4 seminar. See comp1100[1] for course logistics.

Outline

1. Submissions and feedback
2. Minimum viable products — what they're for, the six principles, the types
3. Prototyping — the four axes, paper prototypes, wireframes, Wizard of Oz
4. The economics of design

Minimum viable products

See minimum-viable-products[2] — what an MVP is for (it exists **to test the ability of the product to meet minimal customer needs**, not to earn revenue and not to find bugs), the six principles, the five types, and the prototype-vs-MVP table.

The distinction to hold onto: an MVP is **not a full product, but it is a product, not a prototype**. A prototype is shown to people; an MVP is released to them.

[1] courses/comp1100/comp1100.pdf
[2] courses/comp1100/minimum-viable-products.pdf

> "By far the dominant reason for not releasing sooner was a reluctance to trade the dream of success for the reality of feedback."
>
> — Kent Beck, *Approaching a Minimum Viable Product* (2009)

Chapter 6 of the course notes covers this.

Prototyping

See [prototyping] — prototyping as a form of experimentation inside the design cycle, the four axes (representation, precision, interactivity, evolution), paper prototyping, wireframes, Wizard of Oz, and the economics-of-design cost ladder.

The ladder is the practical takeaway: low fidelity prototypes are *super cheap* and focus on flow and interactivity; high fidelity ones are *quite cheap* and focus on look, feel and usability; code prototypes are *more expensive* and focus on code feasibility; the MVP is *most expensive* and focuses on getting to market. Spend the cheapest amount that answers your question.

Have code generation tools killed wireframing?

An open question, with arguments on both sides — see [prototyping]. The strongest point for "no": wireframes are *intentionally* low fidelity, to get feedback on the user **experience** rather than the user **interface**.

Key takeaways

- MVP = the most important features.
- MVP purpose: **experiment!**
- Simple and easy to build.
- Different MVPs allow us to run different experiments.

Next week

Seminar: 2025-08-25-the-history-of-computing[3] and 2025-08-25-prompt-engineering[4] — the last seminars of the semester.

[3] courses/comp1100/2025-08-25-the-history-of-computing.pdf
[4] courses/comp1100/2025-08-25-prompt-engineering.pdf