Flyp: The (Shorter) Whitepaper

My name is Campbell Nilsen (Twitter/X handle *@nephew_jonathan*); I'm a former Latin teacher and long-term Anki user who, after a few years of teaching, became convinced that a new spaced-repetition software (SRS) platform is desperately necessary. It is well-known that SRS can do amazing work for people willing to put in the time and discipline to use it daily, but that it remains, in the grand scheme of things, niche. Language-learning enthusiast, medical students, and other serious autodidacts represent almost the entire userbase while schoolchildren's grasp of basic content knowledge is at a nadir not seen for decades. The time is ripe for something new, and that is why John Pavlick (*@lambdapriest*) and I are creating Flyp.

The fundamental mechanics of Flyp are little different from those of Anki, Mnemosyne or Supermemo: users create flashcards, which they are tested on at increasingly long intervals to keep them in long-term memory. However, almost everything else has been rethought and updated for the contemporary user. Flyp is in-browser rather than downloadable, which means it can be used on Chromebooks and that there is no need to sync cards across devices. We are aiming for a minimalist, but aesthetically pleasing, appearance (ideally in the spirit of Art Deco or Streamline Moderne) that does not hide key information from the user or require navigating a bewildering array of menus. Automatic transcription is featured for non-Latin scripts—no more copy-pasting Chinese or Arabic—and no knowledge of HTML or Javascript is needed to design card templates. Cards can be set to have other cards as prerequisites, ensuring that material is only reviewed when the basics are mastered, so that the user is building a web of knowledge rather than a motley collection of unconnected facts (see Appendix 1 for an example of what this might look like). An AI assistant can be used to mass-manufacture cards and suggest additions and fixes. Flyp is built for the classroom and for group study just as much as for autodidacts: teachers can make cards for their students, see whether or not they're doing reviews, and see what needs to be re-reviewed in class. (This is reminiscent of Quizlet, but Quizlet is built for cramming rather than long-term retention—it does not have an SRS mode—and its card maker is nowhere near as powerful as Anki's).

The potential market is vast: in addition to the tens or hundreds of thousands of people who use Anki on a daily basis, the 54 million schoolchildren in the US alone represent a ten- or even eleven-figure market. A quick glance at communities such as /r/Anki will show that, as much as its users love it, there is an immense amount of frustration with its unattractive appearance, learning curve, and kludgy mechanics. A platform that can do everything that Anki can do and more, allowing users to upload their existing Anki decks, is one that existing Anki users are well wiling to pay for. Similarly, Quizlet's weak card-making abilities and lack of long-term card tracking represent a real opportunity for a competitor in an age of profound learning loss and missing content knowledge among students. $5-6 per month for individual users, and $20-30 per year per student for institutions, is well in line with establishment platforms such as Quizlet and Mochi but will start printing millions in revenue as soon as it gets traction. (For example, assuming a P/E ratio of 25 and profit margins of 80%, a medium-sized school district with 100,000 middle- and high-school students would represent $40M of valuation alone if paying $25 per student per year; the coming wave of microschools and small charters, whose procurement processes are less complex than those of public school districts, is also an immense opportunity if we can move quickly).

Flyp is still in development (John and I both have day jobs), but progress is rapid and the original white paper (of which this memo is ultimately a summation) received enthusiastic feedback; see appendix 2 for the technical details. We are applying to YCombinator for the January-March 2025 cohort, but do not think we would need the full $500K investment to launch. We believe around $120K would be sufficient to cover enough of our salaries to work on Flyp full time and bring it to usable prototype (John has a family to provide for). We are additionally—and less urgently—seeking a marketing budget of about $200K, which is well more than enough to launch a spate of advertisements near elite schools such as Harvard, Yale and Berkeley (on the theory that there is no ad copy in edtech half as good as 'used by hundreds of Harvard students', and that early traction at elite schools will lead to rapid adoption at their lower-ranked competitors and by K-12 students). Further afield, the international market also presents a lucrative opportunity: nations like India and Vietnam with a cram-heavy educational system and strong cultural emphasis on education hold massive potential even at lower revenue per user.

Because Flyp is a SaaS product, funding is not ultimately necessary to bring it into the world or even to get cash flow going (there are enough fed-up Anki users to bring in monthly revenue in the high four or five figures soon after launch), but it would speed development along immensely and allow us to reach bigger target markets much more quickly. We have incorporated in Delaware as *Flyp, Inc*. and are seeking investment even at this juncture.

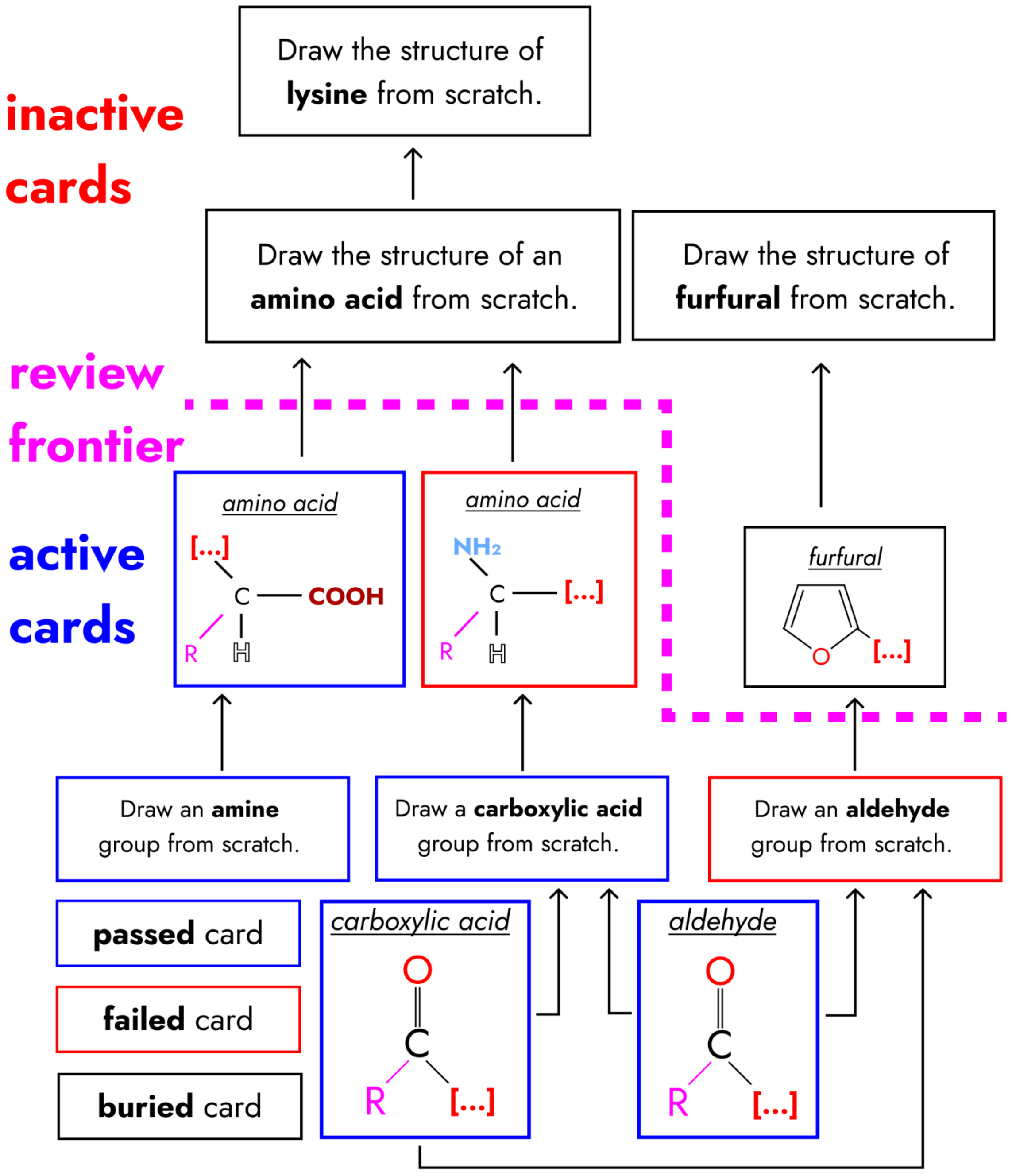
—Campbell Nilsen, CEO
Miami, FL
August 22nd, 2024

campbell@flypcard.com
www.flypcard.com (not much to see yet)
https://x.com/nephew_jonathan

John's contact details:

john@flypcard.com
https://x.com/lambdapriest

Appendix 1: Prerequisites in Flyp


inactive cards
Draw the structure of lysine from scratch.
Draw the structure of an amino acid from scratch.
Draw the structure of furfural from scratch.
review frontier
active cards
amino acid
[...]
C
COOH
R
H
amino acid
NH2
C
[...]
R
H
furfural
O
[...]
Draw an amine group from scratch.
Draw a carboxylic acid group from scratch.
Draw an aldehyde group from scratch.
passed card
failed card
buried card
carboxylic acid
O
C
R
[...]
aldehyde
O
C
R
[...]

This is a partial diagram of a 'web of knowledge' that a Flyp user might build for a bio chemistry class; only prompts are shown (a carboxylic acid group has a hydroxide group, -OH, where an aldehyde has only a hydrogen atom). To my knowledge, no SRS platform on the market can do this.

Appendix 2: Technical Details

We are building Flyp in Elm and running it on Lamdera, a full-stack, hosted Elm runtime environment. Elm and Lamdera are one of the strongest parts of our moat; John has been building in Elm for eight years and is a well-respected community member who currently works for Vendr on one of the world's largest production Elm codebases. Elm's strong and flexible type system, speedy compiler and high-quality feedback loop allow us to ship new features rapidly and refactor old ones safely.

We are optimizing for a lightweight but powerful tech stack to keep technical debt low. Lamdera is a fork of the Elm compiler that only adds what's necessary to support full-stack development while keeping type safety and backend performance. Lamdera's persistence is backed by an immutable log of every event that has been emitted by the application, which provides us with to-the-moment backups and full-stack type-safe migrations.

Lamdera's hosting and deployment platform is part of the compiler and is maintained by the same team; this means that our support agreement covers not just our uptime, disaster recovery, and platform-level issues, but also assistance with application development. The team at Lamdera are all friends of Flyp and want to see us succeed on a personal level; moreover, we have the option of extending the compiler and running everything in-house at any time, minimizing Flyp's dependence on third parties.

This should allow us to run an extremely lean engineering organization well into the future; we suspect that we will never need to hire more than one dedicated employee to work on any part of the organization's technology other than building core application features, and will not need to hire that person (if at all) until we have hit considerable scale.

While all the production code is written by John, I am not devoid of programming skill and still perform a technical role on our team and spend a great deal of time on specification design, flowcharting, pseudocode, and testing. "Build a better Anki" is an easy wish to cast, but a meaningless one without digging into the technical details of what Anki does well, what it could be doing better, and what that looks like on the level of the code.