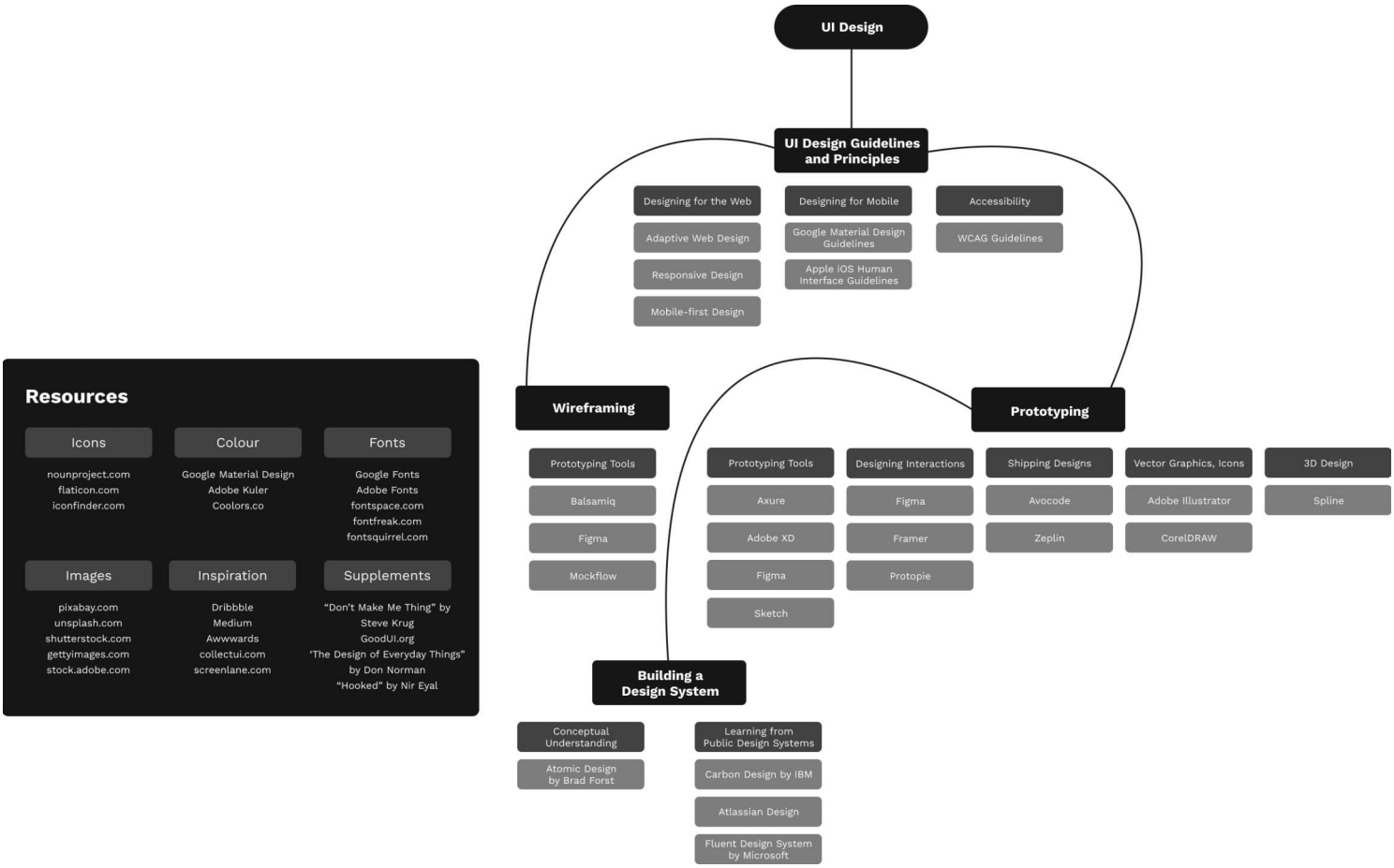
The self-taught UI/UX designer roadmap in 2021

A deep dive guide on teaching yourself UI/UX design from zero knowledge to a full-time role

Andy Chan Follow
Mar 23 · 18 min read ★



September 2020 was when I got hired as a UI/UX designer, even without any experience. Despite being a newbie, I still managed to get a 25% pay raise after three months. Yet, just based on my words alone, it seemed that the journey was smooth-sailing: it was only so because of the many hours put into understanding the ever-evolving UI/UX field.

You might be familiar with the Full-Stack Developer Roadmap, which is a very popular roadmap that describes, ideally, how someone should learn to become a full-stack developer. Based on that idea, I decided to create a UI/UX designer roadmap that precisely suits a self-taught UI/UX designer in 2021 (which is what I went through).

While this guide serves to be definitive, it is still non-exhaustive. Technologies and trends are constantly evolving in the UI/UX field. For example, glassmorphism has emerged as a new trend in UI design in 2020.

Another critical thing to note is that this an essential roadmap that requires autoregulation: you will need to tweak based on the job you're aiming for and your past skillsets. For example, graphic designers can find it easier to learn UI/UX design because of the similarities, but accountants may find it a challenge.

Caveats

Before you go on to learn more about the roadmap, there are some caveats to this roadmap that you should bear in mind while reading:

- **You may find yourself learning more on the job.** There is only so much you can learn without having the opportunity to apply them constantly. While you are learning on the job, you will find that you can make connections between theoretical knowledge and practical applications faster and easier.
- **You don't have to learn everything.** You don't need to know every software, and neither do you need to have every design guideline memorised. Instead, focus on learning *how to learn* and maintain a constant state of curiosity towards all things UI/UX. One trend that you may love today may die out the moment you wake up.
- **Never stop learning.** Ever-changing technologies and design trends mean that you'll constantly learn new things. Ten years back, UI/UX designers didn't need to

care about VR, but there is an increasing demand for UI/UX designers who have done AR/VR apps before.

- **Learn incrementally.** I cannot stress enough about how Rome isn't built in a day. You will eventually gain a wealth of knowledge but learn incrementally. Dissect topics and break them down to study a little bit of it every day, rather than devoting a whole day to it (unless this works for you the best, then that's on you).
- **Be compassionate to yourself.** UI/UX designers, no matter whether they are junior or senior, all go through some form of imposter syndrome. You'll occasionally doubt yourself, frequently second-guess your decisions, and catch yourself wondering if you should've known a piece of knowledge earlier. That's perfectly fine. It's okay to have self-doubt.

Let's jump right into the overall UI/UX Designer Roadmap in 2021.

UI/UX Designer Roadmap in 2021: An Overview

Overall self-taught UI/UX designer roadmap in 2021 (by Andy Chan).

I split the roadmap into three central portions:

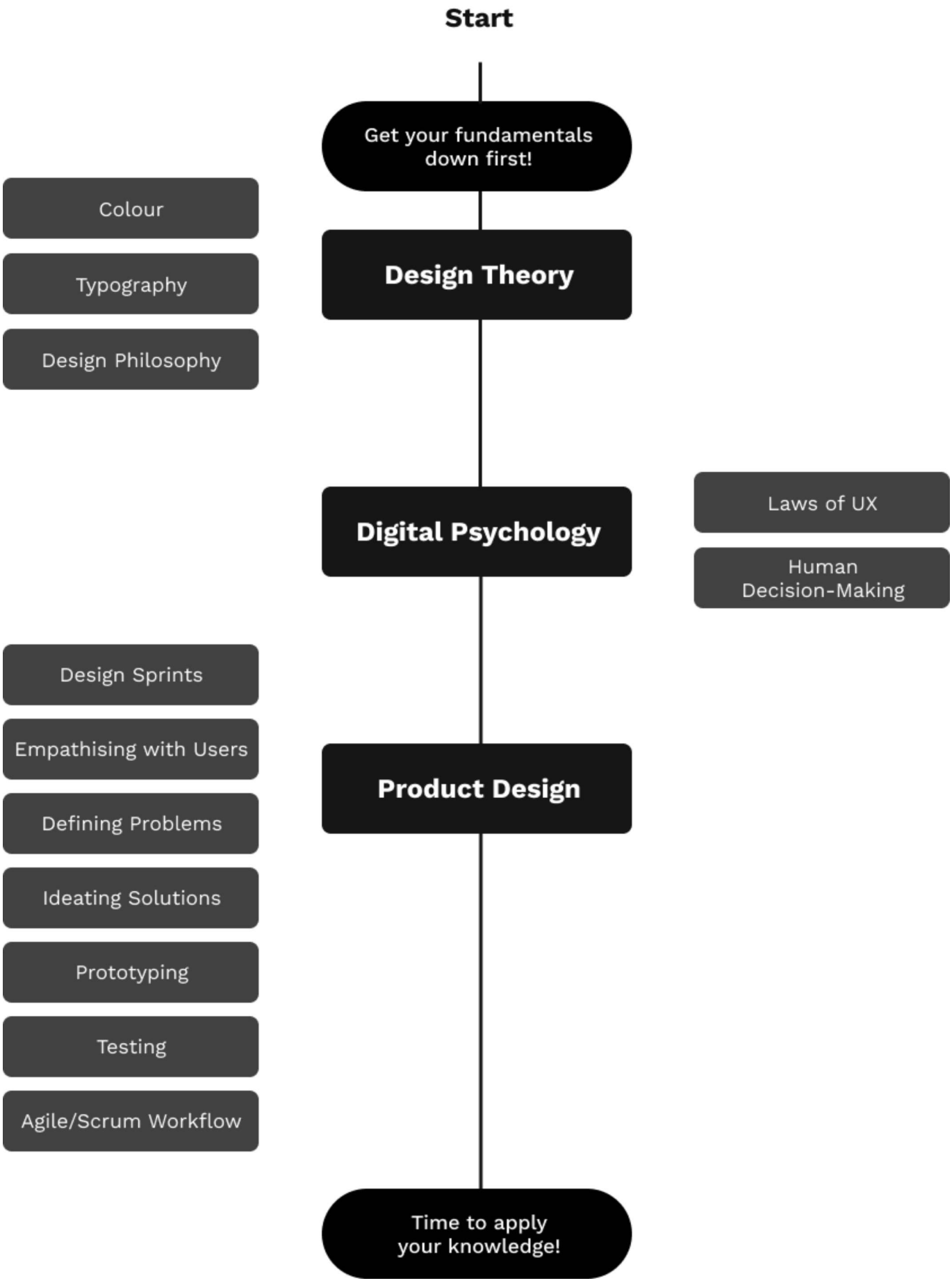
- **Get your fundamentals down first.** This covers design theory to human behaviour. You usually spend about two to three weeks here—realistically, there's way too much knowledge for you to stay at one spot. You ideally just want to know 'enough' for the next steps.
- **Learn your practical skills.** At a junior level, you are most likely not going to specialise in UX or UI unless your employer is a huge company (e.g., Shopee, Amazon). Most of the time, you're looking at UI/UX Designer roles where you're expected to do both. Hence, you should be learning both.
- **Branch and learn other additional skills.** Does UI/UX design end at designing? There's more. I'm a UI/UX designer who does a bit of front-end development, but I

have no-code development experience at a previous stint.

You should be firming up your fundamentals to ensure that you understand why certain decisions are being made in design (e.g., button on the right or the left?). Not everyone has natural design sense, but enough knowledge and practice can help you hone that.

Let's dive into the fundamentals.

What Do I Need for My Fundamentals?


Start
Get your fundamentals down first!
Colour
Typography
Design Philosophy
Design Theory
Digital Psychology
Laws of UX
Human Decision-Making
Design Sprints
Empathising with Users
Defining Problems
Ideating Solutions
Prototyping
Testing
Agile/Scrum Workflow
Product Design
Time to apply your knowledge!

This roadmap assumes that you do not have any knowledge of design. If you are familiar with some things here, feel free to skip them. If not, you can continue reading.

UI/UX designers are the bridge between design and technology. While developers and computer scientists invent new and improve current technologies, your role as the UI/UX designer is to bring their work to the target audience. Think of technology as paint and you being the painter.

Like them, you also need to have a firm grasp of the theoretical knowledge behind your role. Let's start with the first one.

Design Theory

This is pretty straightforward. You are looking at fundamental designs such as colours, typography, shapes, and forms. Here, you're trying to understand the theory that goes behind colour, the terminologies in typography, and training your brain in understanding shapes and forms.

You should not be spending too much time on this as you're not looking to understand it entirely, but rather, understand enough to use it at work.

Digital Psychology

You can also think of this behavioural economics, human behaviour, human digital behaviour, and so on. The core of this topic is to understand how humans act online, why human act in specific ways, and how you can use that to your advantage when designing your product.

A good example would be why big, flashy buttons should be at the bottom of the phone screen. Putting it at the top of the phone screen goes away from the "thumb zone," which means the user has to put slightly more effort into reaching the button. This cognitive load means the user will most likely not press the button at all.

Despite how digital psychology is an essential topic in UI/UX design, it is not something that you have to deep dive into. Knowing surface-level knowledge is enough as you will find yourself understanding more of digital psychology when you get to observe users. Think of this topic as something that you need a certain amount for this period, but you can learn more about it overtime after that.

I broke this down into two main parts:

- **Laws of UX.** Laws of UX is an extremely helpful compendium of best practices in building user interfaces built by Jon Yablonski. It's a good source for you to reference as you design.
- **Human decision-making.** Humans make terrible decisions most of the time, and that's what you need to know. Why do humans make such decisions? Why does indicating a sale when the price has not been reduced compel a human to make a purchase? How do you get humans to be hooked on your app? Understanding some parts of this will help you in the future when you design.

You should spend some time on this before you move on.

Product Design

We're slowly moving away from theory now. After having a solid grasp of design theory and human behaviour, it's time to learn how to design products with that knowledge.

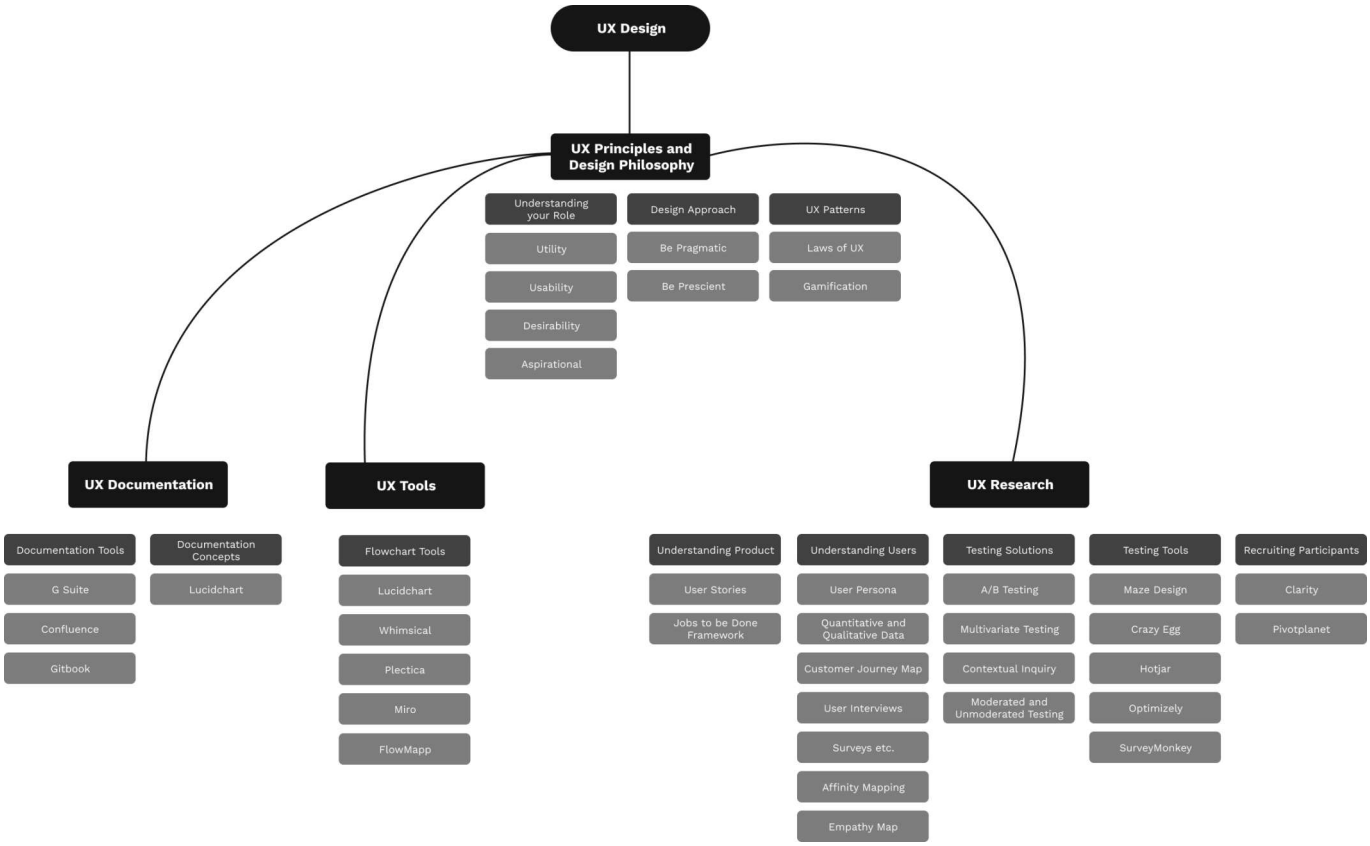
Learning product design helps you to understand how your design can solve problems. While there is a slight difference in product design and UI/UX design, the fundamental idea is to solve user problems. In product design, you're focusing on these things:

- **Design sprints.** How can you generate a massive number of ideas in five days? Take it from Google Ventures.
- **Design thinking.** Design is more than just beautiful user interfaces. It's about understanding your users, knowing what you need to do to solve their problems, then testing your solutions to see if you did. Stanford has a good illustration of what design thinking is.
- **Agile/Scrum workflow.** This depends on whether you're working with engineers that work mainly in Agile/Scrum. It's still good to understand how product design works when you're in an Agile-managed team and what the terminologies are when using such a workflow (e.g., retrospectives, standup, Kanban). Most startups generally use Agile/Scrum workflow.

You shouldn't be spending too much time here as well. While learning about the theory is excellent, most of your knowledge will come from your professional UI/UX design

role. The key here is not to flood yourself with too much information but rather know enough to keep moving.

What do I need for UX Design?



UX Designer Roadmap (by myself)

Whenever I meet someone who wants to learn about UI/UX design, my personal recommendation is to start with UX first—no point learning about how you can build user interfaces when you don't know why your text should be left or middle-aligned. The objective here is to understand four things:

1. You need to design with **guiding principles and philosophy.** This includes adopting a design approach, knowing UX patterns, and understanding who you are as a UX designer.

2. Learn how to conduct **UX research.** Understanding how to do so is the key to getting yourself a job.

3. Know how to **document UX research.** It's an essential skill to learn so much so that I created its own category.

4. Learn how to use **UX tools.** You'll come across many UX tools buy you don't have to master them. You just need to know why you need these tools and then adapt to your company's whichever platform (unless you get to choose).

Guiding Principles and Design Philosophy

As a UX designer, your role is more than just design. You need to consider many aspects of creating experiences. For instance, how does your design make users feel better? How can you use your design to make them feel self-fulfilled?

A four-step framework when designing experiences.

Let's first understand what the typical goals of the UX designer are:

- **Utility.** Is your product/design able to solve the needs of users? For example, you can use the Uber app to book a private hire vehicle to travel to another place. This is

utility.

- **Usability.** Is your product usable? By usable, we are talking about the "degree to which a product can be used." Specifically, are your users able to learn how to use your product with as little friction as possible?
- **Desirability.** Do your users like your product? You don't have to create new mental models, unique needs, or make something unusable. Generally, the design and experience should be likeable on average, such that people don't feel apprehensive about using your product.
- **Aspirational.** A higher-order than desirability, this is where you ask if your users feel good when they use your product. Do you want them to feel safe? Do you want them to feel accomplished?

Every product has its potential to achieve all four, but don't consider them as a hierarchy. While you can achieve a lot of utility, don't forget to consider some usability and desirability. It's more realistic to have a ratio of all four rather than maxing out one element in the set.

By understanding what your role is, you will make better decisions when designing experiences. There are loads of design philosophies out there which you can supplement with books and other resources. Here are some for you to start (there are no affiliate links below):

- "Don't Make Me Think" by Steve Krug
- "The Design of Everyday Things" by Donald Norman
- "Hooked" by Nir Eyal

How to conduct UX Research

Learning how to conduct UX research is the core job scope of the UX designer and product designer. In general, you're learning about your users, then creating designs to test with users to improve the product you're working on. UX research is crucial in creating lasting mental models that hook users onto the product (e.g., TikTok, Instagram).

I won't go too deep into this as this is a vast topic, but I'll cover the topics briefly. Essentially, you need to:

- **Understand the product.** Using user stories and the Jobs-To-Be-Done Framework (JTBD), you can paint a big picture of why people use this product. By understanding the product, you can then understand the nitty-gritty details of the business. This will shape your UX research plans in the future (e.g., more focus on a particular feature = less priority on cosmetic interactions).

- **Understand the users.** This is the first step in the product design process. Empathising with users is extremely important, and there are loads of tools you can use to help you do so. You can gather data using a user interview or a survey, use affinity mapping to cluster important data together, then chart their experience and what they feel at each stage of their interaction with the business using a customer journey map. You can also chart their experience in your product using a UX flowchart. You also need to learn what type of data you're getting, such as psychographic, behavioural, demographic, quantitative, and qualitative data. Having these data will help you ideate solutions later on.

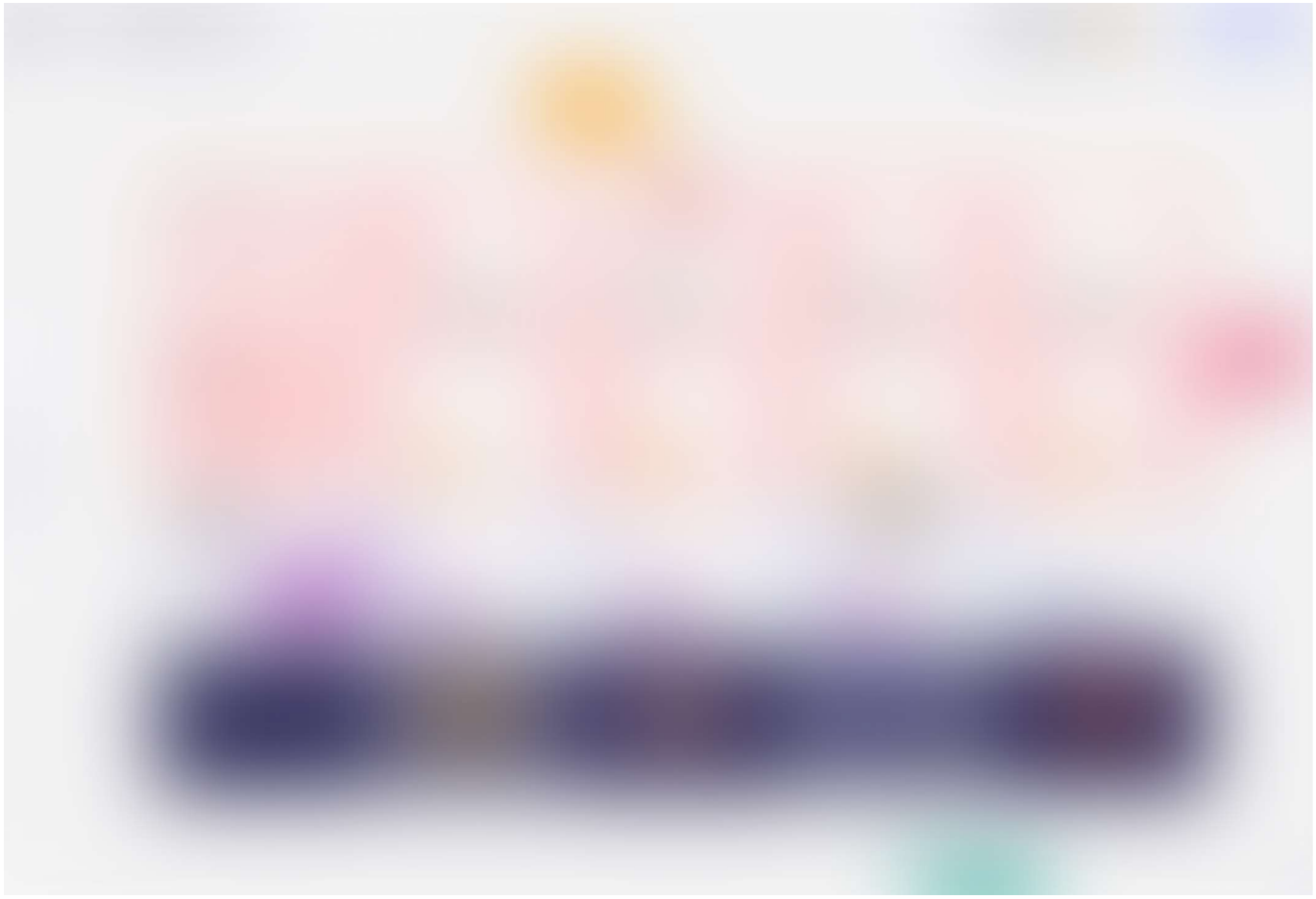
- **Test your solutions.** You ideate solutions, but you can't tell your developers to implement the feature without having some form of evidence. To prove that it works, you will need to recruit participants, set up some form of user interview or contextual inquiry. The key is to note down whether you have proven your hypothesis (i.e., "my feature helps users," "the visual hierarchy of the text is wrong"). Essentially: how do you test your designs, and how do you use those data to help you?

- **Learn to use testing tools.** You need to use tools to test your designs. For instance, Maze Design allows you to test your participants remotely (if that's required). Hotjar is a popular website heatmap tool to track where users click and how long they stay at a particular part of the page, defining your design layout in the future. Either way, don't stick to one testing tool: stick to what works best at that moment.

What UX Tools do I need to use?

You use UX tools to help you visualise and make sense of the data you get from your users. For instance, your customer journey map enables you to visualise a generalised

view of what a customer goes through when using your product to fulfil an objective. UX designers typically use digital tools during their work, and some of them include:



Miro's collaborative platform (src: Miro)

Miro is a tool that I personally use. Think of it as a digital whiteboard that gives you a whole suite of tools to create any chart and diagram you need since it is collaborative (just like Google Documents); you can even use it to conduct activities and workshops.

A visualisation of what Lucidchart can do (src: Lucidchart)

Thinking of using a no-nonsense, simple flowchart tool? Here you have **Lucidchart**, which is also a collaborative diagramming tool. Lucidchart is widely used in the UI/UX industry as well.

Other tools include:

- **Whimsical.** Used by the folks at Notion and Airbnb, Whimsical is a very straightforward collaborative diagramming tool.
- **Plectica.** Used by the folks at Uber and HubSpot, it's a smart, collaborative diagramming tool with nifty features that others don't provide.
- **FlowMapp.** Built mainly for UX designers, folks at Accenture and Intel use it. It offers a more robust toolkit that fits the entire product design process.

Creating UX Documentation

Initially, I thought this to be part of the UX Research category. However, creating UX documentation is of great significance and thus demands its own segment. When it comes to UX documentation, we're looking at documenting our test plans, observations, results, and how that all translates to more revenue for the company.

UX documentation is vital as a form of leverage when you're talking with other departments. For instance, your boss may not want to allocate more budget to your tests

for a feature she thinks isn't that useful. You can also use UX documentation to communicate what you have done and what changes were made to others, especially those who aren't too aware of what y you were doing.

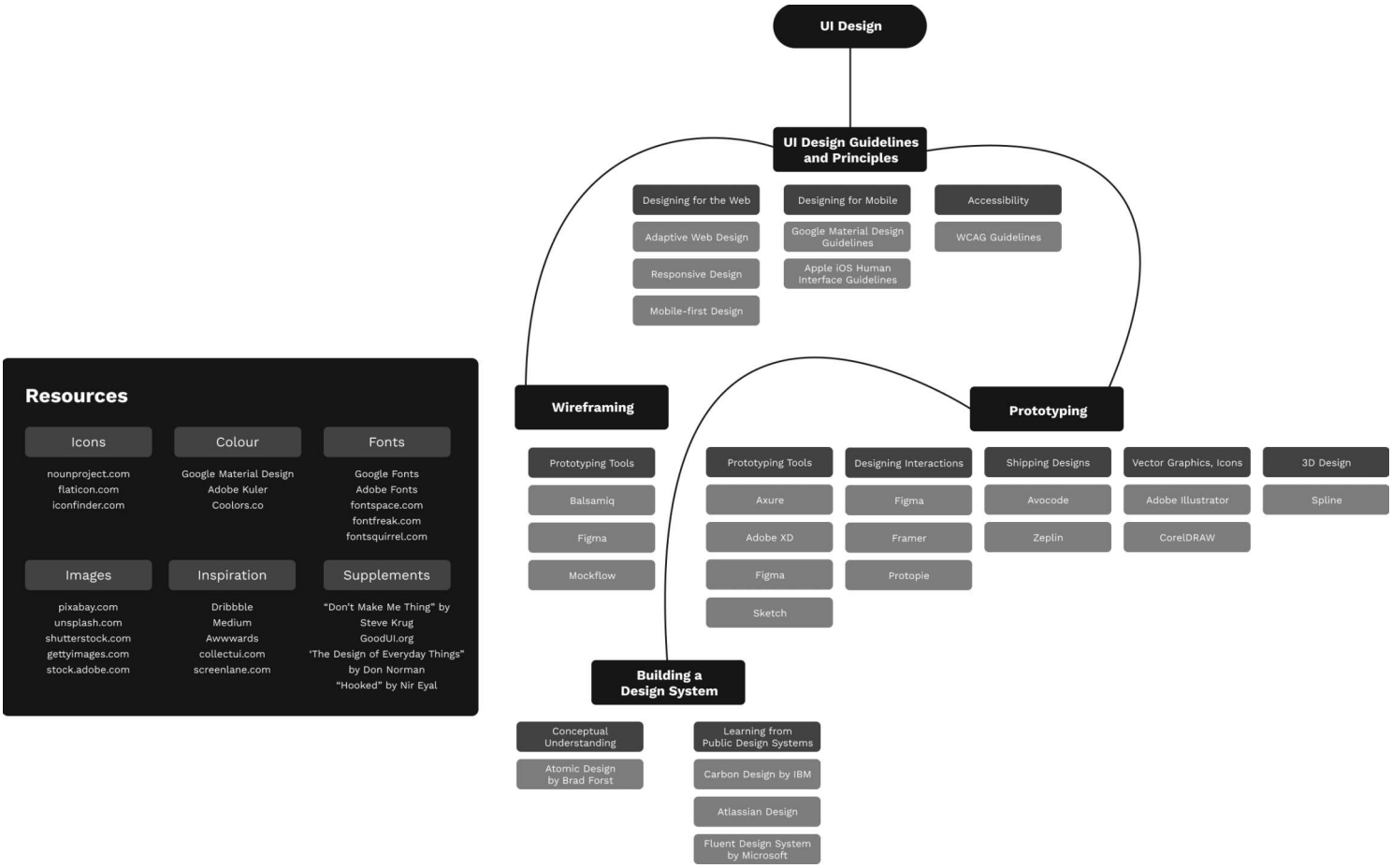
There is no strict convention when it comes to UX documentation, but generally, you should:

- **Communicate your test observations and results.** How did your user test go? What did you do during that user test? How does the result help with your design?
- **Document essential design principles.** What are some design principles you need to evangelise throughout the entire company?

You can document anywhere as long the entire organisation can have ease of access. You can start with Google Documents, but as documentation gets thicker, you can consider other tools like Confluence and Gitbook. Specialised "wiki" tools give you more control over how content is displayed and stored.

Now that we've got the basic layout for UX design down let's dive into UI design.

What do I need for UI Design?



UI Designer Roadmap (by myself)

UI Design Guidelines and Principles

Just like design experiences, you also need guiding principles for user interface designs. Fortunately, this is pretty much laid out for UI/UX designers.

If you're designing for mobile screens, it would be good to check out Google's Material Design for the Android system or Apple's Human Interface Guidelines for the iOS system. Essentially, these are structured content aimed at telling you how to design for a specific operating system, taking note of elements such as typography, colours, and animations.

Apple also laid out the guidelines for designing on MacOS as well. For general browser-based designs, you will need to understand the concepts of responsiveness and mobile-first design.

As with all designs, you need to understand how you can make sure everyone can use your design. Fortunately, we have the W3C, the Web Accessibility Initiative, that came up with accessibility guidelines, also known as WCAG. It is a checklist that allows you to

check if your designs discriminate against anyone with a visual impairment or some form of disability that prevents them from using your design normally. You can also use online tools like a contrast checker to ensure your colour contrasts are clear and that they pass the WCAG guidelines.

Discrimination in design is a widely researched topic, and designers should always strive to ensure that this does not occur.

Wireframing

Wireframing is a way of designing layouts but without all the colours and images. You only want the layout to see the visual structure of your design. It is commonly used in the early stage to establish a basic structure before the content is added. This way, you can quickly adapt the structure if there is an issue with the way things are laid out.

A screenshot of Balsamiq's wireframing tool (src: Noupe)

Wireframing can be done either digitally or on paper. The key is understanding the concept. Once you've grasped the concept of wireframing, you can think about using specific tools for it:

- Balsamiq is a widely used wireframing tool with an easily understandable interface, with a visually pleasing layout for your wireframes such that it looks hand-drawn.
- InVision has a Freeform template that allows you to wireframe reasonably quickly on the platform. However, it's not a platform fully designed for wireframes.

Prototyping

The bread and butter of every UI/UX designer, also known as high-fidelity prototyping. This where you create representations in their closest resemblance to the final design in details, functionality, experiences, interactions and user flow. Developers take your high fidelity prototypes as a point of reference for their work.

A concept design for a logistics app (src: Dribbble)

There are loads of design tools out there for prototyping, and these tools are also evolving rapidly to include new updates. I'll only focus on the ones that are highly popular today:

- Sketch is used by people at Google, Facebook, and Stripe. It's extremely popular, and it is also famous for its plugin marketplace. Sketch can pretty much do everything as long as you have the right plugin.

- Figma is an up and coming design software that is collaborative in nature. It has a desktop app and a web-based browser application. Figma is used by well-known startups such as Github, Dropbox, and Twitter. (I also personally use Figma).

- Framer is another design software that has many convenient features for design systems. It also allows UI/UX designer to create complex interactions. Framer is used by the folks at Microsoft, Spotify and Deliveroo.

- Gravit is a design application that encompasses UI/UX design, vector illustration and even image/photo editing. It is a complete design software for all your needs without going into the Adobe Creative Cloud.

That's not the end for prototyping: you may also have to use software like Avocode to ship your designs. Newer software like Spline is also coming up with 3D design tools that software like Figma and Sketch cannot provide.

Regardless of what software you use, you mustn't get too attached to a single tool at the start. However, you should not download all of them and create prototypes on all of them. Instead, download them, try them all out, and pick the one you think is the most suitable or comfortable for you.

If your future employer requires you to use another platform, it is okay: you will most likely learn on the job anyway.

Building a Design System

What's a design system? A visual way to explain this would be to check Atlassian Design or IBM's Carbon Design. Imagine if your login page, your sign up page, and your homepage all looked highly different from one another. That would be a disaster in terms of consistency, and design systems are meant to ensure the branding and style is consistent throughout.

When doing UI design, you'll likely work with existing design systems, enhance existing design systems or even build your own. This is where the understanding of Atomic

Design comes in. Conceptually, all design systems are similar. However, creating them and using them depends on the software you are using.

What else is there?

You've done it—that's all there is to UI/UX design… or is it?

There's no end to the number of things you can learn, and like in software development, you don't have to know everything. Over time, you will determine whether you are more suited to be a UX designer or a UI designer. Perhaps you may love Sketch more than Figma. Your favourite way to document UX research maybe something like Notion. The key is always to remain open-minded: nothing is set in stone, even human behaviour.

With that said, UI/UX designers do get the option to branch out and learn other skills that make themselves more attractive as hires or to venture into new roles.

Front-end Development

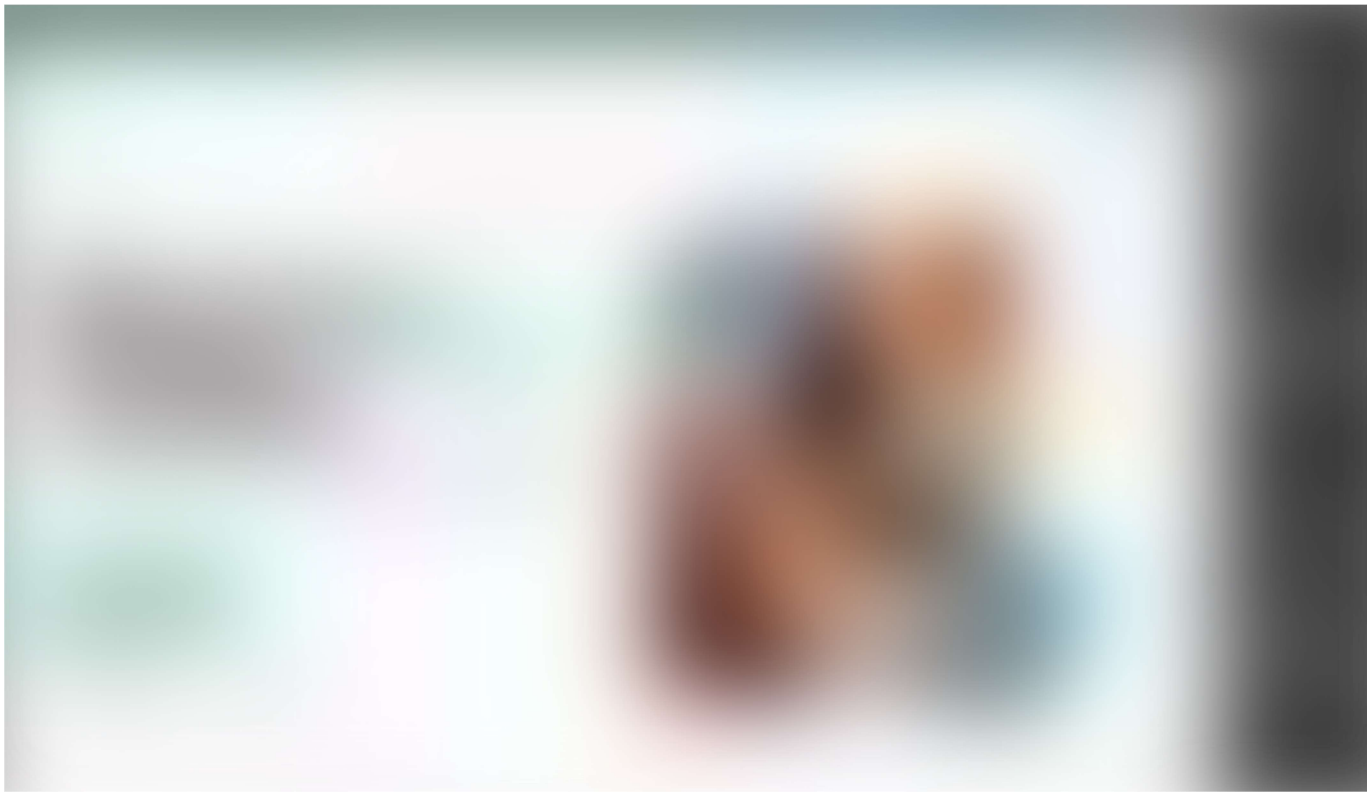
Some UI/UX designers like myself are also front-end developers. Essentially, these are UI/UX designers who also have the skill to code user interfaces. While they are also able to understand databases and manage them, it is not their speciality. Building prototypes using ReactJS and NodeJS is more of their jam. You can check out this popular front-end developer roadmap to know how you can start being a front-end developer.

You don't need to be a front-end developer to keep your UI/UX job, though. It's perfectly fine to be an expert in UI/UX design with just an understanding of what goes on behind the scenes. Personally, I stress that UI/UX designers should proactively learn about front-end development even if they are not good at it. Learning about CSS and HTML basics can help you converse with your developer colleagues much easier.

No-code Development

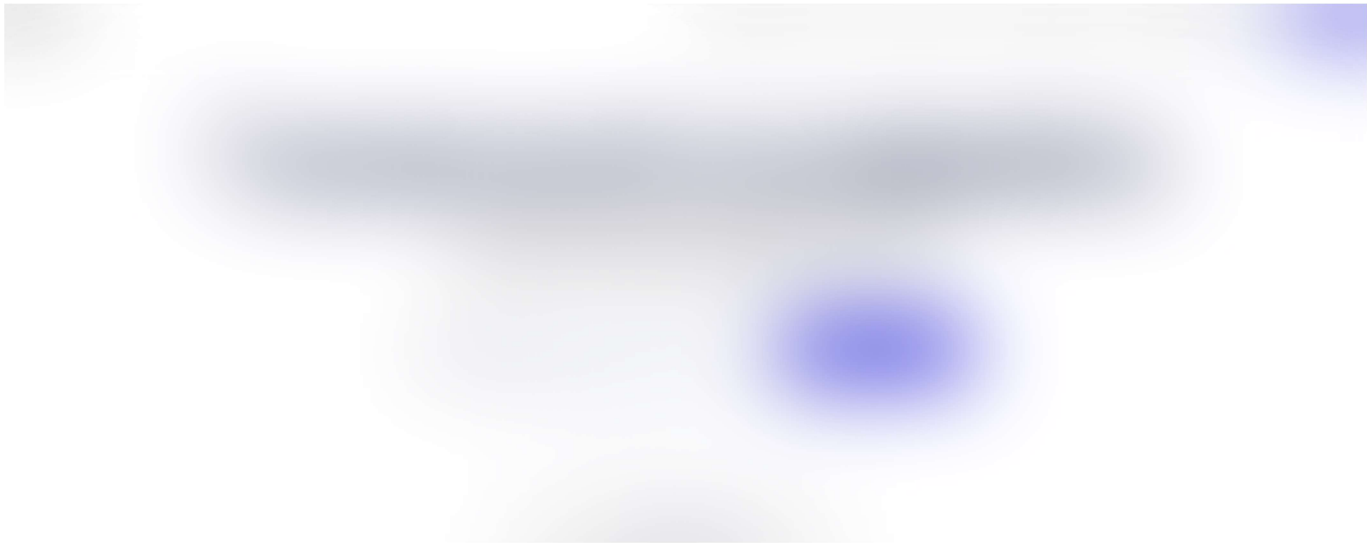
Website builders like Wix and Weebly are no longer the market leaders. Instead, you have more complex yet more powerful web applications that allow you to build modern, sleek websites and mobile applications.

No-code development is basically using no-code development platforms to do rapid prototyping. This is a handy skill to learn. As a UI/UX designer, you can quickly prototype your designs (beyond just a design prototype that does not receive real data) without asking your developers to collaborate with you on a test. Here are some tools that are on the rise today:



Screenshot of Webflow's internal site editor (src: Webflow)

Webflow is all the rage these days, and for a good reason: their application is powerful enough to build sites like this, this, this, and this.



Screenshot of Bubble's website (src: Bubble)

Bubble is a powerful web-app builder. Using zero code, you can make complex logic using Bubble. Some startups use Bubble to create their software and subsequently raised funding like Teeming and Qoins.

There are even more no-code development platforms out there, some even specifically for mobile apps.

Product Management

Besides those two, UI/UX designers can also transform themselves into product managers. Because of prior experience in customer scoping and creating wireframes, UI/UX designers can transfer their skillsets over to product management. Here's a good article on how this transition works and what was the difference.

Regardless of how you learn UI/UX design, the road to being a "good" UI/UX designer is always rocky. Most of the time, you'll find yourself learning things on the job rather than from formal education or online courses.

The core of a UI/UX designer is to create solutions to solve user problems, and that's all you need to care about.

Thanks to Fabricio Teixeira.

UX Design UX Design UI Technology