

Identification et choix d'une stack technologique

<https://github.com/heig-vd-mvp-course>

[Web](#) • [PDF](#)

L. Delafontaine et V. Guidoux, avec l'aide de GitHub Copilot

Ce travail est sous licence [CC BY-SA 4.0](#).

Plus de détails dans le support de cours

Cette présentation est un résumé du support de cours. Pour plus de détails, consultez le [support de cours](#).

Objectifs (1/3)

- Décrire ce qu'est une stack technologique web
- Identifier les avantages et les inconvénients d'une stack technologique web
- Évaluer le choix d'une stack technologique web par rapport à une situation donnée



Objectifs (2/3)

- Énumérer des stacks technologiques web populaires
- Décrire ce qu'est une landing page
- Expliquer l'intérêt d'une landing page avec formulaire de contact pour un MVP
- Décrire comment créer une landing page



Objectifs (3/3)

- Décrire où trouver des templates de landing page
- Décrire où héberger une landing page avec formulaire de contact
- Décrire comment héberger une landing page avec formulaire de contact
- Lister les éléments clés du référencement en lien avec les moteurs de recherche



Choisir la stack technologique adaptée à votre projet

- Étape cruciale pour le succès de votre projet (performance, maintenabilité et évolutivité)
- Détermine les technologies à utiliser pour le développement
- Dépendant de chaque projet



Un petit retour dans le passé

- Afin de bien comprendre les enjeux actuels, il est important de comprendre l'histoire du web :
 - Les débuts du web
 - L'explosion du web
 - La révolution des smartphones
 - Indexation et référencement



Les débuts du web

- Le web a été créé en 1989
- Réservé aux universités et aux laboratoires de recherche
- Nombre d'utilisateurs très faible
- De simples pages HTML avec du CSS et du JavaScript
- Un nombre très variés de site web différents (peu de standards)



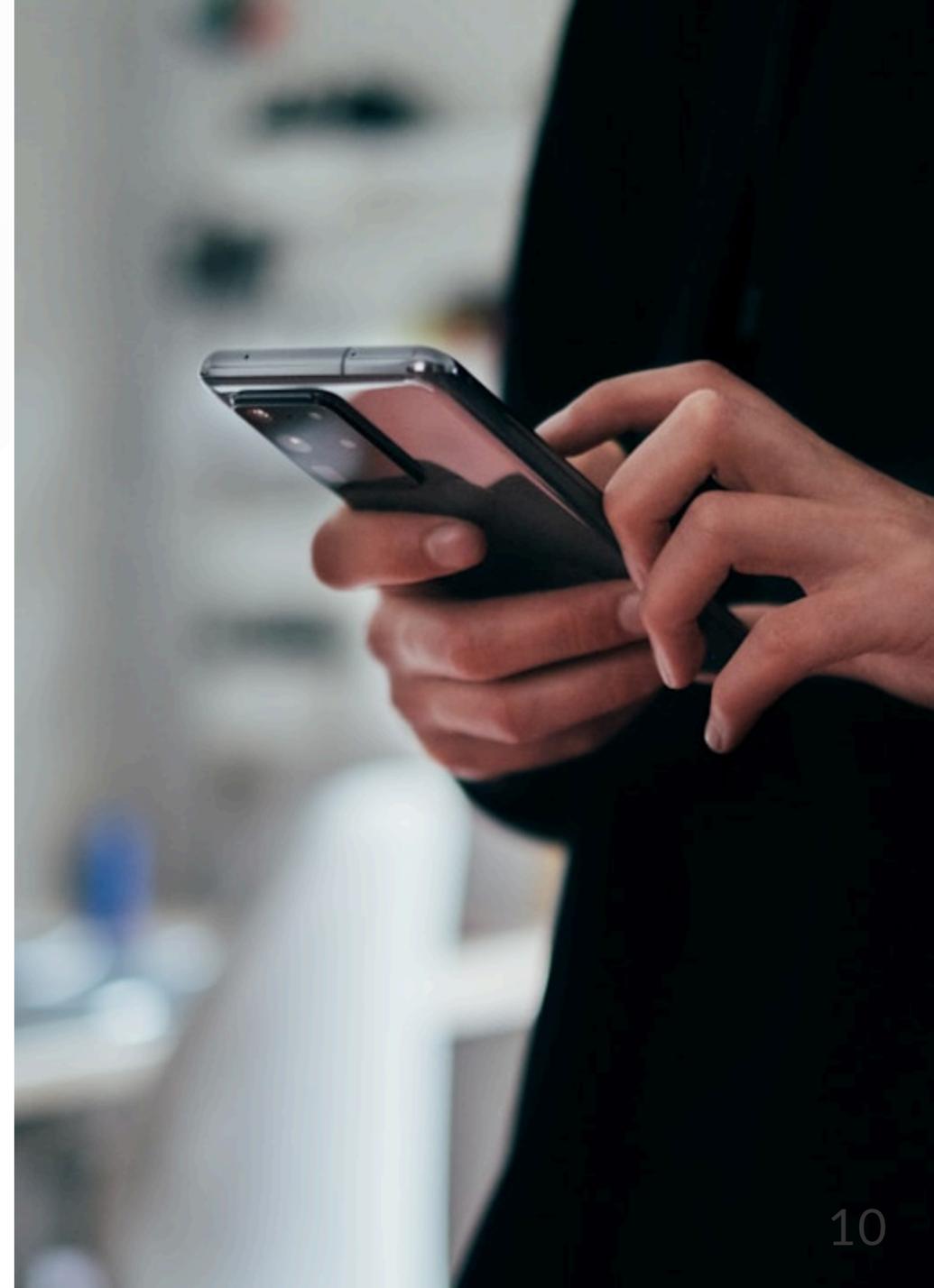
L'explosion du web

- L'arrivée du grand public sur le web
- Le nombre d'utilisateurs explose
- Les sites web deviennent plus complexes
- Les standards se mettent en place
- Majoritairement des sites web rendus côté serveur (Server-side rendering (SSR)) - PHP, Ruby, Python, etc.



La révolution des smartphones

- L'arrivée des smartphones a tout changé : les sites web doivent proposer la même expérience (réactifs, responsive, etc.)
- Le client est responsable de la génération du contenu (Client-side rendering (CSR)) - JavaScript, Angular, React, Vue.js, etc.



Indexation et référencement

- D'autres problématiques sont apparues : l'indexation et le référencement
- On revient à un rendu côté serveur avec des solutions hybrides (Server-side rendering (SSR) + Client-side rendering (CSR))



Comprendre ce qu'est une stack technologique web (1/2)

Un ensemble d'outils, de langages, de frameworks et de conventions utilisés pour construire une application web.

Du code source à la production, elle comprend généralement :

- Des outils de versioning
- Des outils de collaboration
- De la documentation
- De la sécurité
- Un ou des langages de programmation

Comprendre ce qu'est une stack technologique web (2/2)

- Des outils de développement
- Un ou des frameworks
- Une ou des bases de données
- Des tests unitaires, d'intégration et de bout en bout
- Des outils de revue de qualité de code
- Des outils de CI/CD
- Des services de déploiement dans le cloud
- Des outils de livraison/déploiement
- Des outils de monitoring

Avantages et inconvénients d'une stack technologique web

Avantages

- Accessibilité en tout temps
- Facilité de mise à jour
- Fonctionne sur quasi tous les appareils

Inconvénients

- Gestion des données en ligne et hors ligne
- Sécurité (accès et rôles)
- Performances
- Compatibilité entre les navigateurs

Choisir la stack technologique adaptée à votre projet (1/3)

Le monde du web bouge extrêmement vite ; il est parfois difficile de choisir la meilleure stack technologique pour son projet. Voici quelques questions à se poser :

- Quels sont les besoins ?
- Quel est votre budget ?
- Quelles sont vos compétences ?
- Quelle est la taille de votre équipe ?
- Quelle est la durée de vie du projet ?

Choisir la stack technologique adaptée à votre projet (2/3)

Le choix d'une stack technologique dépend de nombreux facteurs et peut varier en fonction de la situation.

Une fois un projet démarré, il est difficile de changer de stack technologique.

Les sections qui suivent vous aideront à choisir la stack technologique la plus adaptée à votre projet. Il s'agit bien de réponses subjectives.

Choisir la stack technologique adaptée à votre projet (3/3)

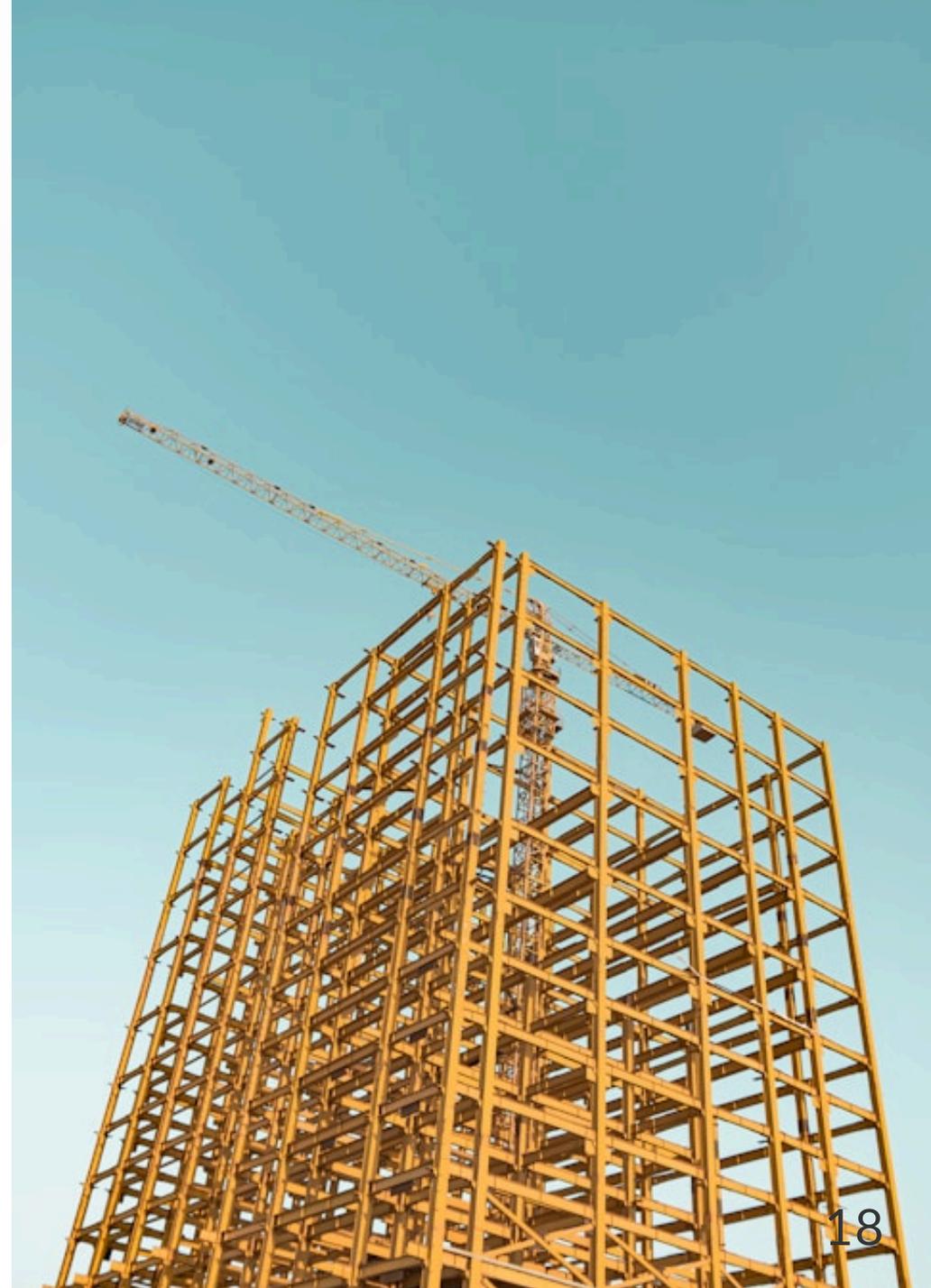
“ Il n'y a pas de solution miracle : chaque projet est unique et nécessite une approche adaptée.

Tout est question de compromis.

Si c'était aussi simple, tout le monde utiliserait la même stack technologique de la même manière et il n'y aurait pas de débat. ”

Quel(s) langage(s) de programmation/framework(s)/outil(s) choisir

- Chacun a ses avantages et inconvénients
- Partez avec celui que vous connaissez le mieux, sinon avec celui qui est le plus populaire
- Il n'y a pas de "meilleur" choix



Solutions "*faites maison*" vs solutions clé en main

- Plus rapide et sûr de partir d'une solution clé en main
- Avec une solution "*faite maison*", que se passe-t-il si la personne quitte l'entreprise ?
- Les solutions "*faites maison*" ne devraient être utilisées que pour se former et apprendre



Full-stack vs frontend/backend

- Dépend de vos compétences, de vos besoins et de vos intérêts
- La solution full-stack est souvent plus simple à mettre en place
- La séparation frontend/backend permet une meilleure séparation
- Les deux approches se valent



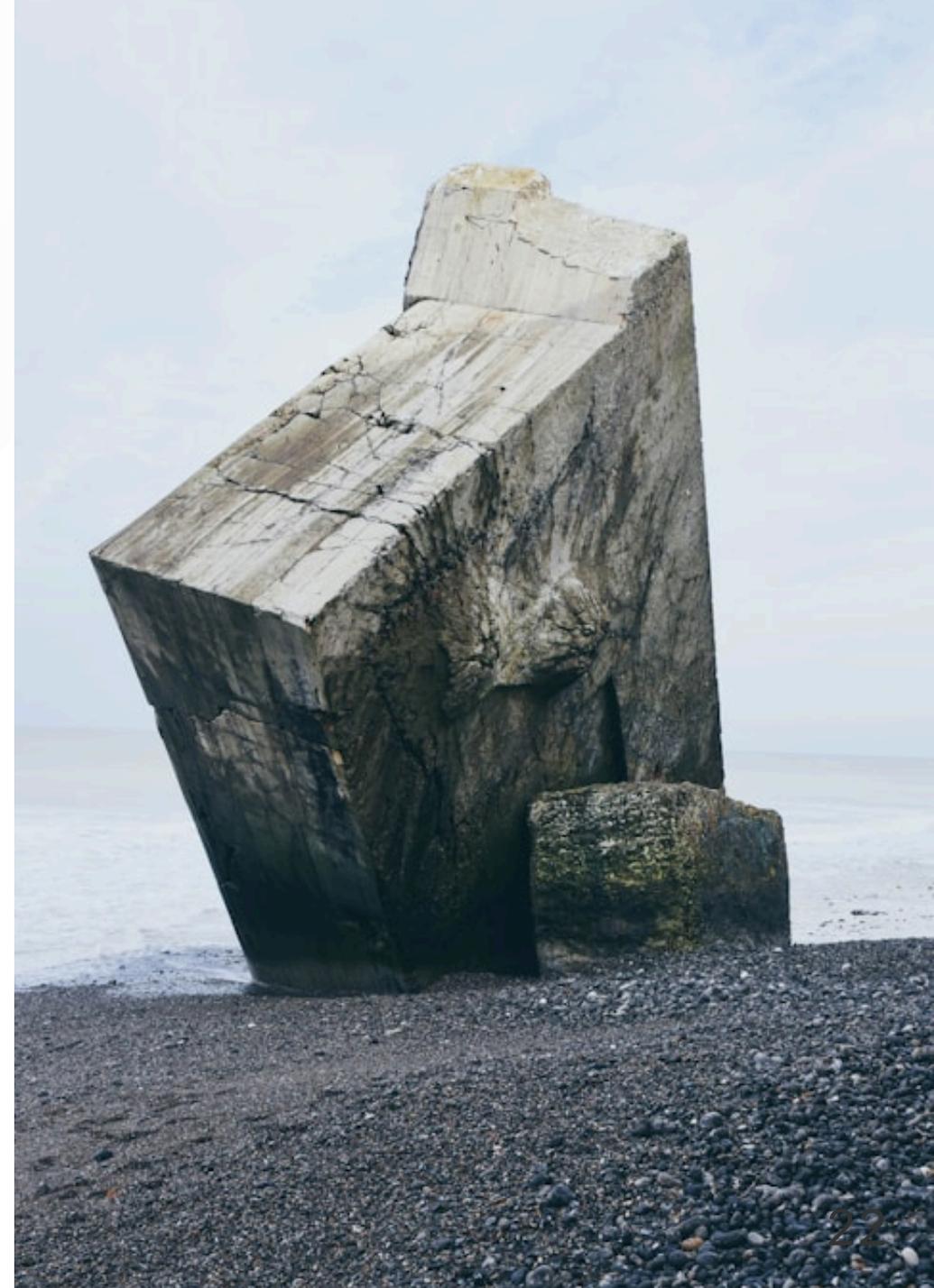
Performances vs maintenabilité

- Nous considérons que la maintenabilité est plus importante que les performances
- Un projet est souvent amené à évoluer et à être maintenu
- Commencez par avoir un produit qui fonctionne et ensuite améliorez les performances



Monolithique vs microservices

- Un monolithe est une application développée en un seul bloc
- Un microservice est une application développée en plusieurs blocs qui communiquent entre eux
- Les microservices sont plus complexes et nécessitent plus de ressources (surtout humaine)



Polyrepo vs monorepo

- Un polyrepo est un dépôt par projet
- Un monorepo est un dépôt pour tous les projets
- Les monorepos sont plus simples à gérer
- Les polyrepos sont plus simples à mettre en place
- Les monorepos sont plus simples à maintenir



SQL vs NoSQL

- SQL est plus adapté pour les données structurées
- NoSQL est plus adapté pour les données non structurées
- SQL est plus simple à mettre en place et à maintenir
- NoSQL est plus complexe à maintenir et finissent souvent par être des "poubelles à données"



"boring technologies" vs "shiny new things"

- Il est tentant de vouloir utiliser les dernières technologies
- Ce n'est pas toujours la meilleure solution
- Optez pour des solutions stables et maintenues
- D'autres personnes vont reprendre votre code



Exemples populaires de stacks technologiques web

- Laravel/Symfony pour PHP
- Ruby on Rails pour Ruby
- Django pour Python
- Spring Boot/Quarkus pour Java
- NestJS/Adonis.js pour Node.js
- React/Vue/Svelte pour le frontend
- Next.js/Nuxt/SvelteKit pour le frontend (et le backend)

Conclusions

- Facebook a commencé avec PHP
- GitHub et GitLab avec Ruby on Rails
- Il n'y a pas de solution miracle, tout est une question de compromis
- Les buzzwords répondent souvent à des besoins ultra spécifiques
- La technologie n'est qu'un outil
- Commencez simple et évoluez au fur et à mesure que votre produit et votre équipe grandit

Créer une landing page efficace pour votre MVP

Définition d'une landing page

- Page web dédiée à une action précise (inscription, contact, achat, etc.)
- Conçue pour optimiser la conversion
- Épurée et orientée vers un objectif unique



Intérêt d'une landing page pour votre MVP

- Valider rapidement l'intérêt pour votre produit/service
- Collecter efficacement les données utilisateur
- Limiter les coûts et efforts initiaux
- Faciliter le dialogue direct avec les premiers utilisateurs



Créer, déployer et maintenir une landing page efficacement

- De simples fichiers HTML, CSS, JavaScript, etc. peuvent suffire
- Parfois chronophage et besoin de compétences techniques (visuel)
- Il existe des outils et des services pour ça



Générateurs de sites statiques pour créer une landing page

Un générateur de sites statiques permet de créer des sites web à l'aide de fichiers Markdown. Cela permet d'avoir un site web léger, rapide et facile à maintenir. Les outils suivants sont populaires :

- [Hugo](#) (+1)
- [Jekyll](#)
- [Gatsby](#)
- [VuePress](#)
- [Astro](#)
- <https://jamstack.org/generators/>
- <https://github.com/myles/awesome-static-generators>

Trouver des templates pour votre landing page

Des templates pré-faits peuvent aider pour un résultat esthétique :

- [Hugo Themes](#)
- [HTML5 UP](#)
- [Bootstrap](#)
- [One Page Love](#)



Ajouter un formulaire de contact à votre landing page

Comme un site statique n'a pas de backend, il est difficile de gérer des formulaires. Des services existent pour gérer les formulaires :

- [web3forms](#)
- [Google Forms](#)
- [Formspree](#)
- [FormBackend](#)
- [Netlify Forms](#)

Cela permet de collecter des retours et des données utilisateurs.

Hébergement d'une landing page

Solutions pratiques d'hébergement rapide :

- **GitHub Pages** : gratuit, adapté aux pages statiques
- **GitLab Pages** : similaire à GitHub Pages
- **Netlify** : simple, rapide, gratuit, idéal pour pages avec formulaires
- **Firebase Hosting** : facile à configurer si vous utilisez Firebase
- **Heroku** : adapté aux applications nécessitant un backend
- **Vercel** : performant pour les applications modernes

Déployer un site statique sur GitHub Pages

Étapes principales pour déployer un site statique sur GitHub :

1. Créez un nouveau dépôt sur GitHub
2. Ajoutez votre code HTML, CSS et JavaScript ou utilisez un générateur de site statique
3. Validez les changements et poussez-les sur GitHub
4. Activez GitHub Pages dans les paramètres du dépôt (GitHub Actions ou fichier CNAME)
5. Configurez le domaine personnalisé si nécessaire

Bonnes pratiques de référencement pour votre landing page

- Utilisez des mots-clés pertinents
- Optimisez les balises `<title>` et `<meta>`
- Créez des URL propres et des balises `<h*>` pertinentes et hiérarchisées
- Faites attention aux performances et à l'accessibilité (balises sémantiques, images, etc.)
- Utilisez <https://pagespeed.web.dev/> pour valider

Faire indexer votre landing page par les moteurs de recherche

- Utilisez Google Search Console pour soumettre votre page
- Créez un fichier `robots.txt` pour indiquer les pages à indexer
- C'est un processus continu : surveillez

Questions

Est-ce que vous avez des questions ?

À vous de jouer

- Prendre connaissance du [support de cours](#)
- Poser des questions si nécessaire



Sources (1/3)

- [Illustration principale](#) par [Viktor Forgacs](#) sur [Unsplash](#)
- [Illustration](#) par [Aline de Nadai](#) sur [Unsplash](#)
- [Illustration](#) par [Nathan Dumlao](#) sur [Unsplash](#)
- [Illustration](#) par [Aron Visuals](#) sur [Unsplash](#)
- [Illustration](#) par [Coolcaesar](#) sur [Wikipedia](#)
- [Illustration](#) par [Taylor Vick](#) sur [Unsplash](#)
- [Illustration](#) par [Jonas Leupe](#) sur [Unsplash](#)
- [Illustration](#) par [Maksym Kaharlytskyi](#) sur [Unsplash](#)

Sources (2/3)

- [Illustration](#) par [Ashkan Forouzani](#) sur [Unsplash](#)
- [Illustration](#) par [Nadya Spetnitskaya](#) sur [Unsplash](#)
- [Illustration](#) par [Fotis Fotopoulos](#) sur [Unsplash](#)
- [Illustration](#) par [Jonathan Chng](#) sur [Unsplash](#)
- [Illustration](#) par [Jonathan Chng](#) sur [Unsplash](#)
- [Illustration](#) par [Emile Guillemot](#) sur [Unsplash](#)
- [Illustration](#) par [Eric Prouzet](#) sur [Unsplash](#)
- [Illustration](#) par [Sunder Muthukumaran](#) sur [Unsplash](#)

Sources (3/3)

- [Illustration](#) par [Nechama Lock](#) sur [Unsplash](#)
- [Illustration](#) par [Rob Laughter](#) sur [Unsplash](#)
- [Illustration](#) par [Jana Niggeloh](#) sur [Unsplash](#)
- [Illustration](#) par [Steven Wright](#) sur [Unsplash](#)
- [Illustration](#) par [Nikita Kachanovsky](#) sur [Unsplash](#)