



# Introduction to open-Source Software (OSS)

Concepts, strategies, and methodologies  
related to open-source software development

Week 03 – Lecture 06



Jamil Hussain

jamil@sejong.ac.kr

010-6252-8807

**Office:** 421, Innovation Center  
Sejong University

# Recap



- Aspects of Licenses
- The GPL and License Compatibility
- Choosing a License
- The community and its structure

# Today, Agenda



- Getting start the OSS project
- Ingredients for starting new project
  - Naming and branding your project
  - Have a clear mission statement
  - State that the project is free
  - Features and requirements list
  - Development status
  - Downloads
  - Version control and bug tracker access
  - Communications channels
  - Developer guidelines
  - Documentation
  - Choosing a license and applying it

# Starting an Open Source Project



<https://opensource.guide/starting-a-project/>

# First, Look Around for same problem

프로젝트는 항상 무→유 x. 인터넷에 코드 찾아보기.

- Always look around to see if there's an existing project that does what you want
- No reason for you to reinvent the wheel again
- But generally, there's no point not looking, and the payoff can be huge.
- If the usual Internet search engines don't turn up anything, try searching directly on



<https://github.com/>



<https://freshcode.club>



<https://openhub.net>



<https://directory.fsf.org>

# First, Look Around for same problem

- Even if you don't find exactly what you were looking for, you might find something so close that it makes more sense to join that project and add functionality than to start from scratch yourself.

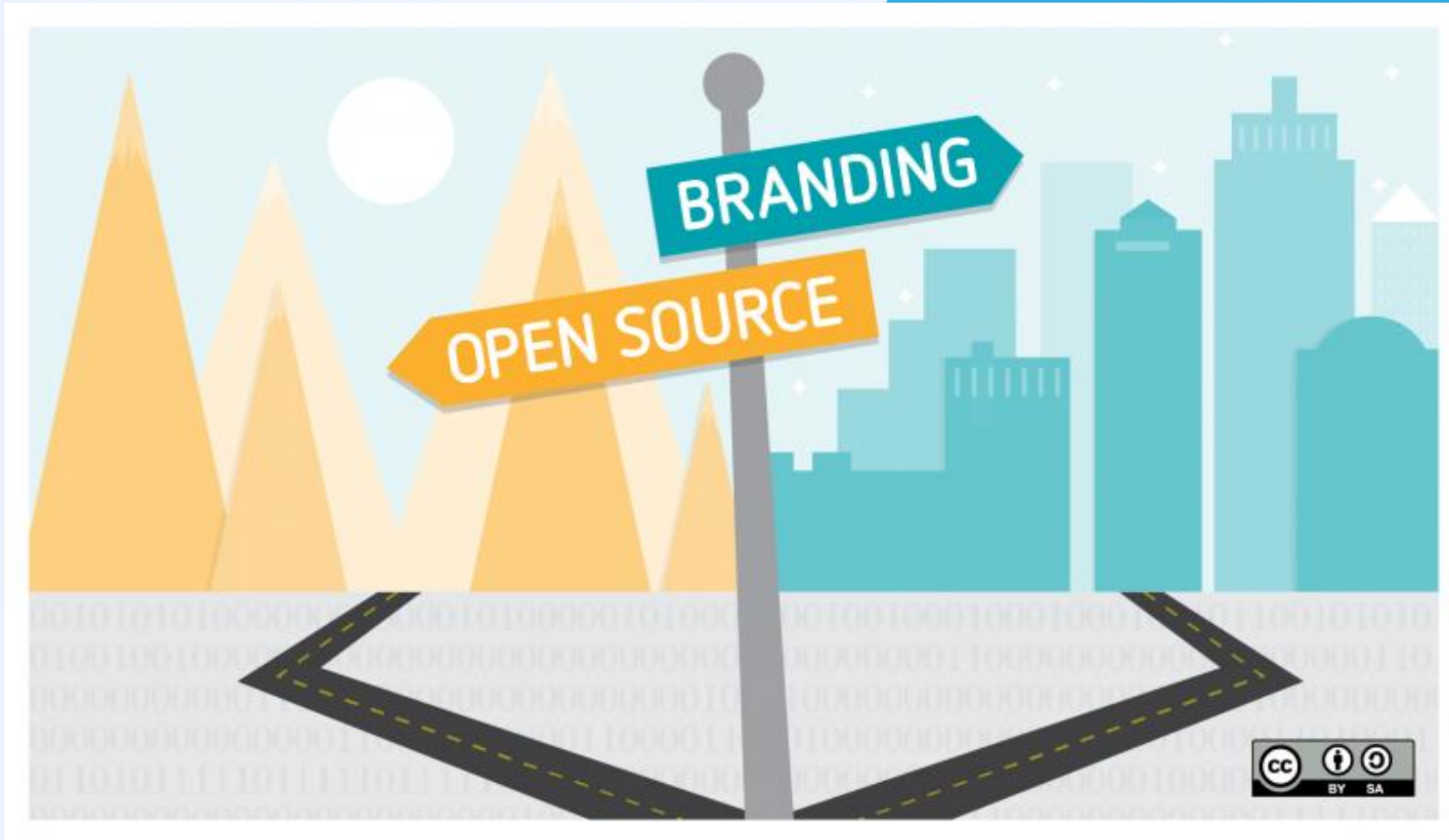
# Starting From What You Have

- Transforming a private vision into a public one
- You or your organization may know perfectly well what you want, but expressing that goal comprehensibly to the world is a fair amount of work.
- You and the other founders must decide what the project is really about
  - that is, decide its limitations, what it won't do as well as what it will
  - and write up a mission statement.
- What makes it so laborious is that it consists mainly of organizing and documenting things everyone already knows
  - "everyone", that is, who's been involved in the project so far.

# FOSS project key items list

- ✓ Naming and branding your project
- ✓ Have a clear mission statement
- ✓ State that the project is free
- ✓ Features and requirements list
- ✓ Development status
- ✓ Downloads
- ✓ Version control and bug tracker access
- ✓ Communications channels
- ✓ Developer guidelines
- ✓ Documentation
- ✓ Choosing a license and applying it





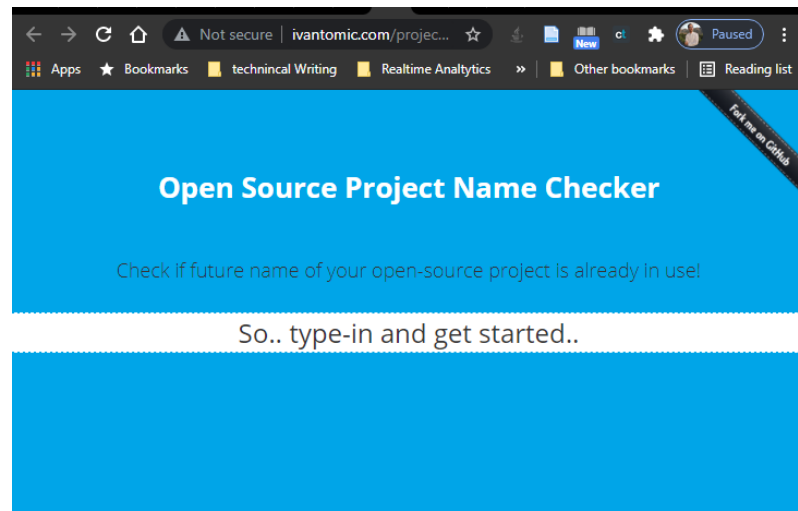
# Naming and branding your project

# Naming and branding your project

- Choosing the good/right name
  - Pick a name that is easy to remember and, ideally, gives some idea of what the project does. For example:
    - [Thin](#) is a fast and simple Ruby web server
  - If possible, is available as a domain name in the .com, .net, and .org top-level domains.
  - If possible, is available as a username on <https://twitter.com/> and other microblog sites.
  - If you're building upon an existing project, using their name as a prefix can help clarify what your project does (for example, node-fetch brings window.fetch to Node.js).

# Naming and branding your project

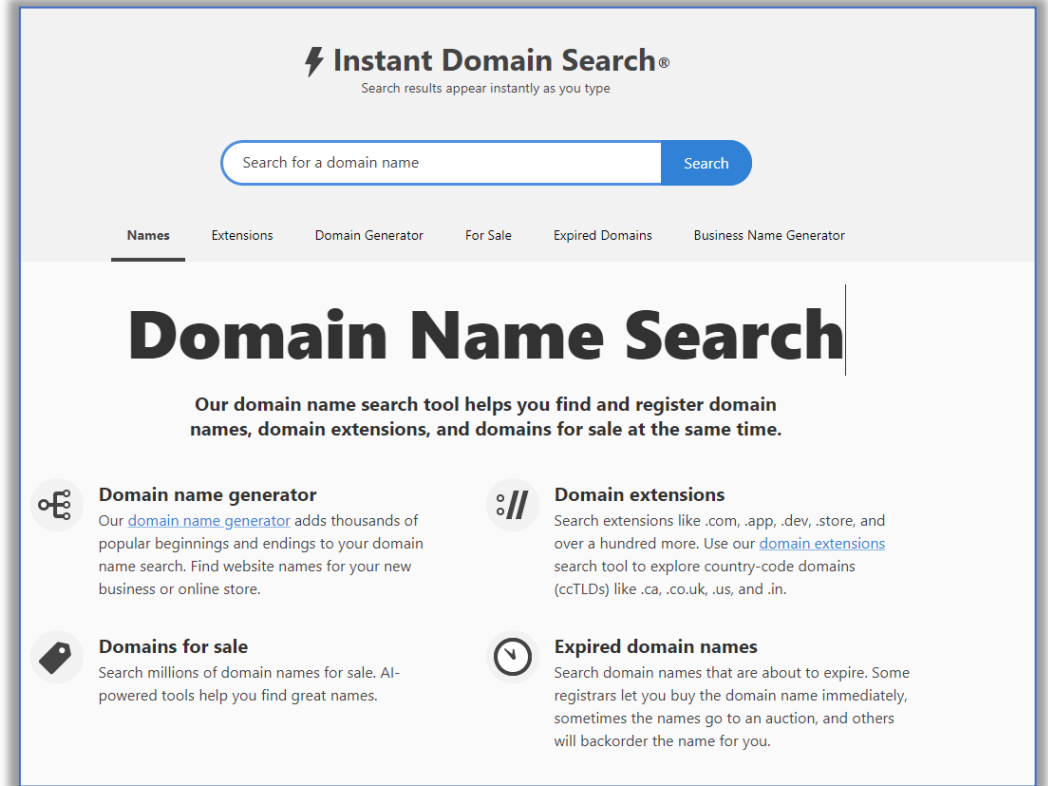
- Avoiding name conflicts
  - Is not the same as some other project's name, and does not infringe on any trademarks.
  - Check for open source projects with a similar name, especially if you share the same language or ecosystem. If your name overlaps with a popular existing project, you might confuse your audience.



<http://ivantomic.com/projects/ospnc/>

# Own the Name in the Important Namespaces

- For large projects, it is a good idea to own the project's name in as many of the relevant namespaces on the Internet as you can.
- If you have the same name in all the places where people would look for you, you make it easier for people to sustain a mild interest in the project until they're ready to become more involved.

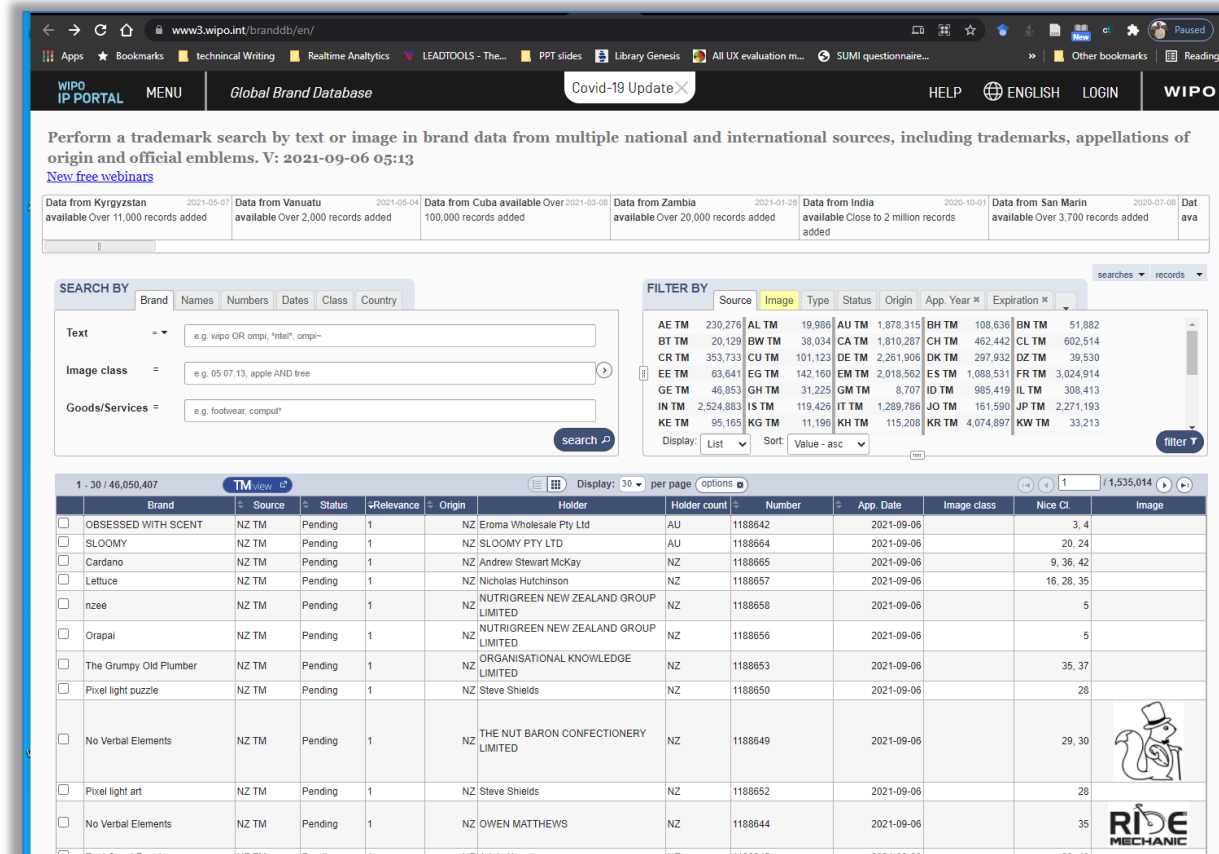


<https://instantdomainsearch.com/>

+ GoDaddy, Cafe24

# Trademark Search

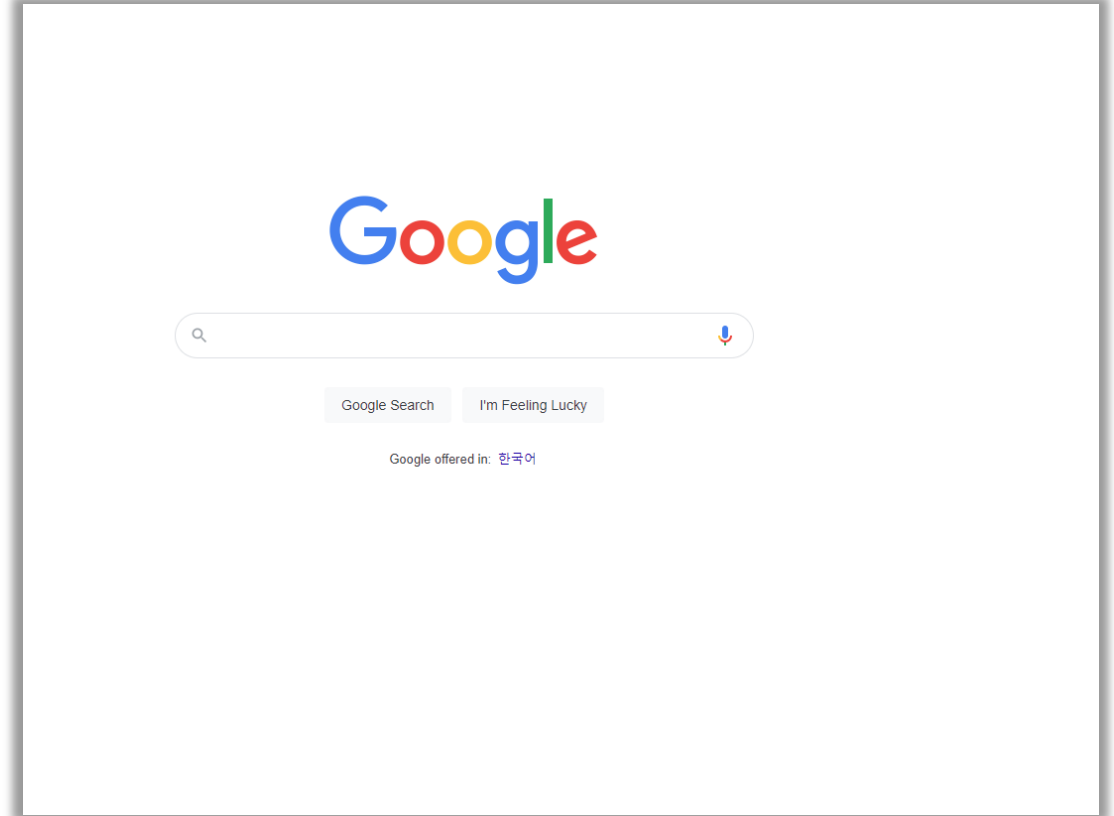
- Make sure that your project's name doesn't infringe upon any trademarks. A company might ask you to take down your project later on, or even take legal action against you. It's just not worth the risk.
- You can check the [WIPO Global Brand Database](#) for trademark conflicts. If you're at a company, this is one of the things your legal team can help you with.



<http://www.wipo.int/branddb/en/>

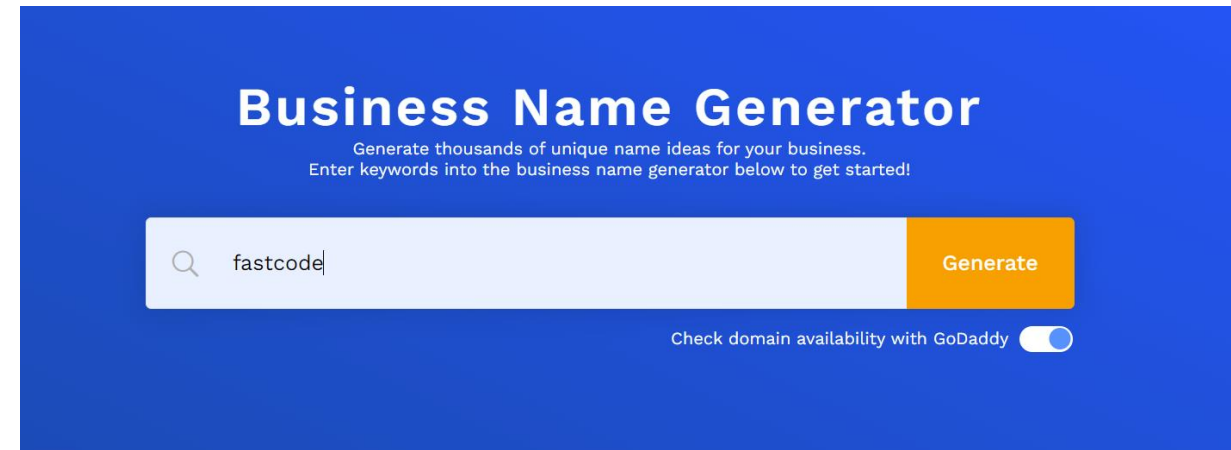
# Google search for name

- Finally, do a quick Google search for your project name. Will people be able to easily find your project? Does something else appear in the search results that you wouldn't want them to see?



# Auto Generate Name

- Generate thousands of unique name ideas for your open source project.
- Enter keywords into the business name generator below to get started!
- Other Options
  - <https://online-generator.com/name-generator/project-name-generator.php>
  - <https://wpdonuts.com/the-best-domain-name-generators/>

The image shows a screenshot of the 'Business Name Generator' website. The page has a blue background. At the top, the title 'Business Name Generator' is displayed in white. Below the title, there is a subtitle: 'Generate thousands of unique name ideas for your business. Enter keywords into the business name generator below to get started!'. In the center, there is a search bar with a magnifying glass icon on the left and the text 'fastcode' inside. To the right of the search bar is an orange button labeled 'Generate'. Below the search bar, there is a toggle switch for 'Check domain availability with GoDaddy', which is currently turned on.

<https://businessnamegenerator.com/>

**Q4:** If the “UXMastermind” name is not available, generate another good using auto-generate name tools and share the suggested name





# Have a Clear Mission Statement

- Once they've found the project's home site, the next thing people will look for is a quick description or mission statement, so
- they can decide (within 30 seconds) whether or not they're



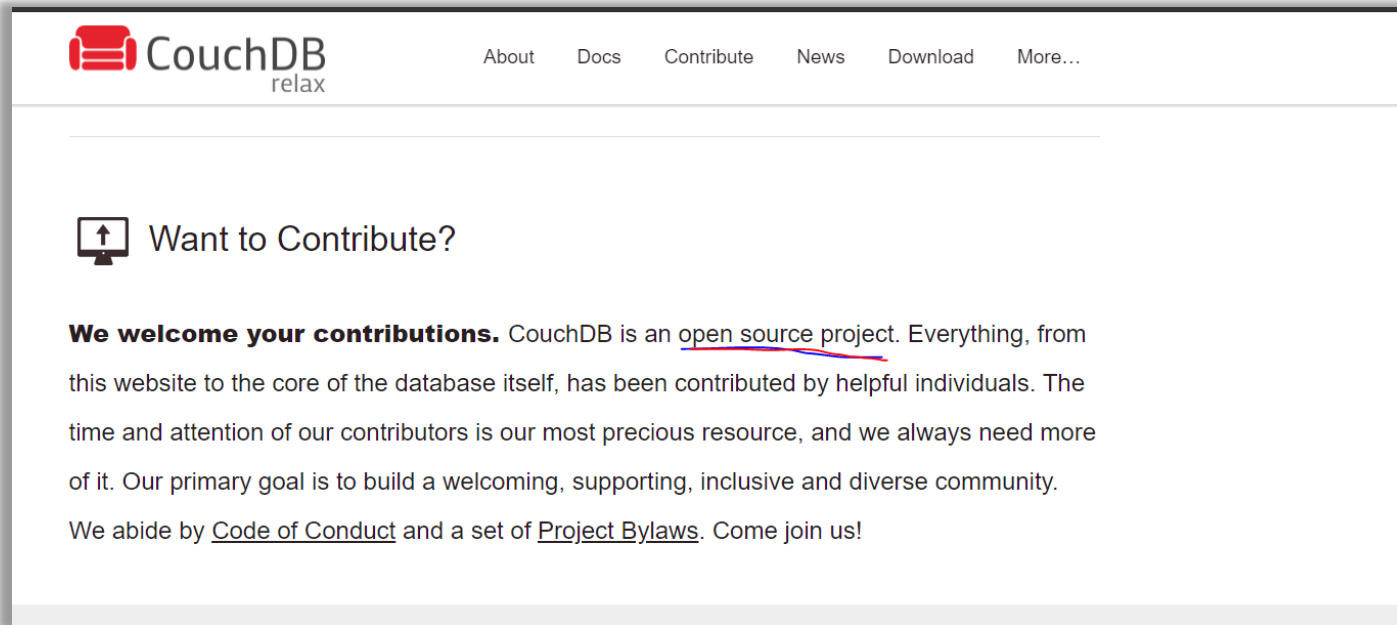
The Apache™ Hadoop® project develops open-source software for reliable, scalable, distributed computing.

The Apache Hadoop software library is a framework that allows for the distributed processing of large data sets across clusters of computers using simple programming models. It is designed to scale up from single servers to thousands of machines, each offering local computation and storage. Rather than rely on hardware to deliver high-availability, the library itself is designed to detect and handle failures at the application layer, so delivering a highly-available service on top of a cluster of computers, each of which may be prone to failures.

<https://hadoop.apache.org>

# State That the Project is Free

- The front page must make it unambiguously clear that the project is open source



<http://couchdb.apache.org/>

# Features and Requirements List

- There should be a brief list of the features the software supports
  - (if something isn't completed yet, you can still list it, but put "planned" or "in progress" next to it), and
  - the kind of computing environment required to run the software
- Think of the features/requirements list as what you would give to someone asking for a quick summary of the software
- It is often just a logical expansion of the mission statement.
- The features and requirements list would give the details, clarifying the mission statement's scope
- With this information, readers can quickly get a feel for whether this software
  - has any hope of working for them, and
  - they can consider getting involved as developers too.

# Features and Requirements List

- Features:

- Searches plain text, HTML, and XML
- Word or phrase searching
- (planned) Fuzzy matching
- (planned) Incremental updating of indexes
- (planned) Indexing of remote web sites

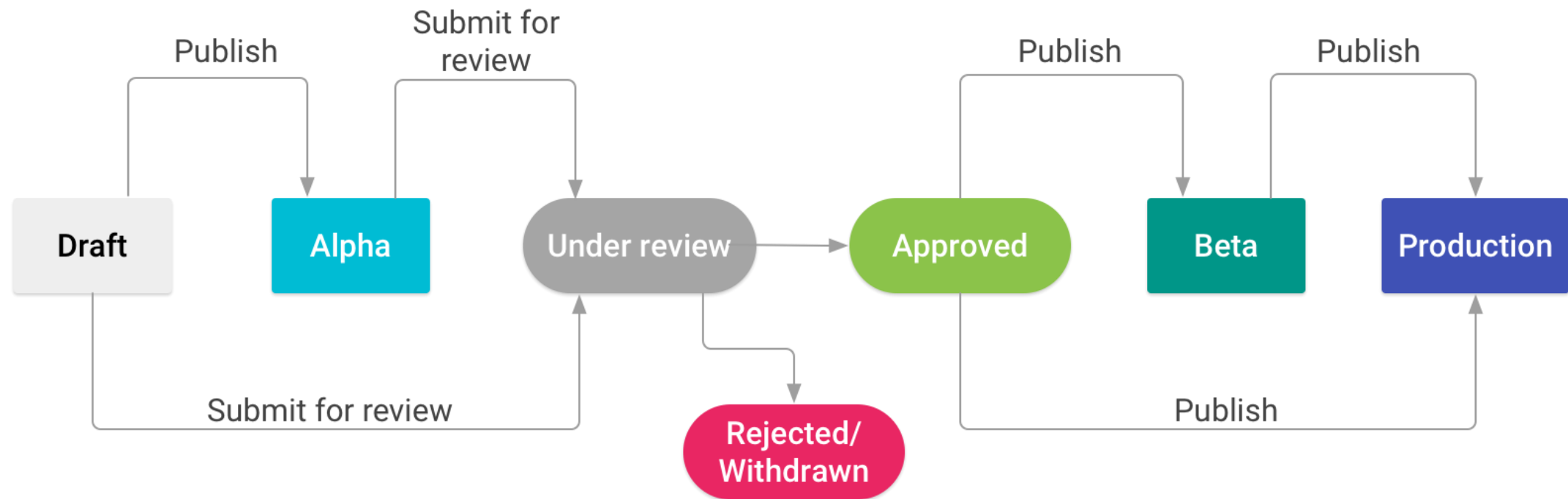
- Requirements:

- Python 3.2 or higher
- Enough disk space to hold the indexes (approximately 2x original data size)

# Development Status

- Visitors usually want to know how a project is doing. For new projects, they want to know the gap between the project's promise and current reality.
- For mature projects, they want to know how actively it is maintained, how often it puts out new releases, how responsive it is likely to be to bug reports, etc.
- Development Status Should Always Reflect Reality
  - Alpha and Beta

# Alpha and Beta



# Downloads

- The software should be downloadable as source code in standard formats
- When a project is first getting started, binary (executable) packages are not necessary, unless the software has such complicated build requirements or dependencies that merely getting it to run would be a lot of work for most people.
- (But if this is the case, the project is going to have a hard time attracting developers anyway!)

# Version Control and Bug Tracker Access

- Downloading source packages is fine for those who just want to install and use the software, but it's not enough for those who want to debug or add new features.
- People need real-time access to the latest sources, and a way to submit changes based on those sources.
- The solution is to use a version control system
  - Specifically, an online, publicly-accessible version controlled repository, from which anyone can check out the project's materials and subsequently get updates.



# Communications Channels

- Visitors usually want to know how to reach the human beings involved with the project
- Provide the addresses of mailing lists, chat rooms, IRC channels and any other forums where others involved with the software can be reached.
- Make it clear that you and the other authors of the project are subscribed to these mailing lists, so people see there's a way to give feedback that will reach the developers.
- Your presence on the lists does not imply a commitment to answer all questions or implement all feature requests.

# Developer Guidelines

- If someone is considering contributing to the project, she'll look for developer guidelines.
- Developer guidelines are not so much technical as social:
  - they explain how the developers interact with each other and with the users, and ultimately how things get done.

# Documentation

- Documentation is essential. There needs to be something for people to read, even if it's rudimentary and incomplete.
- Documentation should be available from two places: online (directly from the web site), and in the downloadable distribution of the software
- For online documentation, make sure that there is a link that brings up the entire documentation in one HTML page
  - (put a note like "monolithic" or "all-in-one" or "single large page" next to the link, so people know that it might take a while to load).
  - This is useful because people often want to search for a specific word or phrase across the entire documentation.

# Documentation

- But this is not necessarily the most common way documentation is accessed.
- Often, someone who is basically familiar with the software is coming back to search for a specific word or phrase, and to fail to provide them with a single, searchable document would only make their lives harder.



**Read *the* Docs**

# Hosting

- Where on the Internet should you put the project's materials?
- A web site, obviously — but the full answer is a little more complicated than that.



bluehost

HostGator

inmotion.  
hosting

GoDaddy

TSOHOST

HOSTINGER

SiteGround

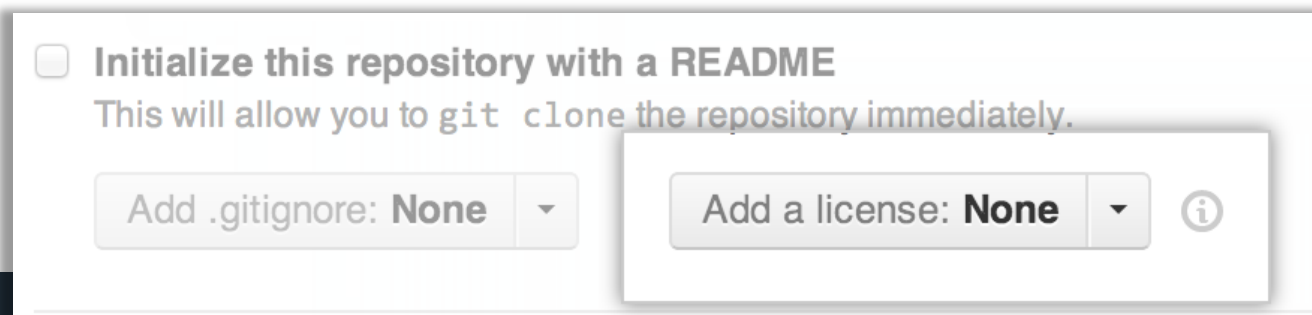
Hostwinds

W

WIX

# Choosing a License and Applying It

- An open source license guarantees that others can use, copy, modify, and contribute back to your project without repercussions. It also protects you from sticky legal situations. You must include a license when you launch an open source project.
- [MIT](#), [Apache 2.0](#), and [GPLv3](#) are the most popular open source licenses, but [there are other options](#) to choose from.
- When you create a new project on GitHub, you are given the option to select a license. Including an open source license will make your GitHub project open source.



☐ **Initialize this repository with a README**  
This will allow you to `git clone` the repository immediately.

Add .gitignore: **None** ▼

Add a license: **None** ▼ ⓘ

# Reading Materials

- Karl Fogel, *Producing Open Source Software: How to Run a Successful Free Software Project*, O'Reilly Media, 2009.
- <https://choosealicense.com/>
- <https://opensource.guide/starting-a-project/>

# Thanks

Office Time: Monday-Friday (1000 - 1800)

You can send me an email for meeting, or any sort of discussion related to class matters.

[jamil@sejong.ac.kr](mailto:jamil@sejong.ac.kr)