Nicolas Bayona

Systems engineering student passionate about technological stuff and the use of technology to improve people's lives. You can check my **LinkedIn** profile here.

Education

2019-2023 (expected):
BSc, Systems engineering; Pontifical Xaverian University (Bogota, Colombia)

- Final project title: Still not sure :C

Experience

Volunteer Work Experience:

Academic mentor at Pontifical Xaverian University:

- As an academic mentor at the university I was in charge of:
  - Welcoming new coming students to a new stage in their lives.
  - Helping the new coming students in their adaptation process.
  - Tell students about my experience as an student so far, my greatest challenges and important lessons I learnt throughout my life.
  - Act as a leader to new coming students, so that they felt they were accompanied in their adaptation processes.

Technical Experience

Most of my technical work is in my GitHub profile, you can visit my profile at **github.com/nclsbayona**

My Cool Side Projects:
Usually I practice my skills via some helpful project that I tend to mantain over time, I think it's really good to automate repetitive tasks and jobs a bot or some type of automation software can do on its own. Examples of this are my

- These items can also contain lists, but you need to mind the indentation levels in the markdown source.
- Second item.

Open Source: * ***Project 1:*** Here

- ***Project 2:*** Description
- ***Project 3:*** We both

Programming Languages:
Throughout my life, I've worked in many projects and therefore, I've adquired a lot of experience in various programming languages. Next, I'll name a few and some relevant stuff I learnt from them.

- ***C++:***
  The first programming language I learnt was C++, the most valuable thing I think I learnt from

my C++ learning was programming logic. I think that C++ is a really great language because it acts as a basis for other languages and many tools that have been working flawlessly since it first appeared almost 40 years ago.

  - *Advantages:*
    - C++ is really efficient in terms of memory consumption, execution time, and comparatively in energy consumption thanks to being a compiled language.
    - C++ allows the use of not only classes but c-like structs too, which gives the language a lot of possibilities and extensibility too.
  - *Disadvantages:*
    - C++ is a complex language, not only because of its syntax but also because of other stuff like pointers and security, and therefore, it requires a lot of time to implement algorithms in C++.
    - Due to C++ being a compiled language, executables are strictly attached to an specific architecture and platform-dependent, which implies that an executable file might present problems when being executed in a platform or an architecture different from the one it was compiled in the first place.
    - There exist libraries that rely on a specific platform and its capabilities, and that might not work in an environment different from the one it was developed in first place, this makes the development process more tedious and so, its not always possible to use the language for a specific problem/task.

- ***Python:***

  After learning C++, I decided to learn Python, because its reputation was great, it simplified the development process as its syntax was similar to natural language and included really powerful mathematical tools that helped to perform tasks that might involve mathematical stuff.

  - *Advantages:*
    - Python has a lot of cool modules that help in a lot of situations and extend the language base capabilities making it easier to perform multiple tasks.
    - The Python ecosystem is completely open-source, which implies that tools and modules that use Python, are open-source as well, which makes everything more transparent and allows users to be able to contribute with fixes and improvements.
    - The syntax Python uses is similar to natural language, which makes the development process comparatively faster.
    - Python allows applications to be mostly platform-independent, because there might exist modules that don't work in all platforms, and therefore, a file can be developed once and used in multiple, different platforms without any changes to the source code.
  - *Disadvantages:*
    - Python is not always as efficient as it can be. This is mainly because it is an interpreted language and so, there exist a lot of tasks that need to be done in execution time different from executing an instruction.
    - Since Python is not a compiled language, a program might not be run completely, which can cause some problems with data consistency and integrity.
    - There exist libraries that rely on a specific platform and its capabilities, and that might not work in an environment different from the one it was developed in first place, this makes the development process more tedious and so, its not always possible to use the language for a specific problem/task.

- ***Java:***

  We both know this one's pushing it.

  - *Advantages:*
    * .
  - *Disadvantages:*
    * .
- ***Other relevant information:***
  Also, I have basic knowledge of other programming languages like ***TypeScript***, ***JavaScript***, ***C***, ***Dart*** *and* ***Swift***, as well as markup languages such as ***YAML***, ***HTML***, , ***Markdown***, ***XML*** *and* ***JSON***, and other tech-related tools like ***Git***, **Kubernetes - Kubectl**, **Docker - Podman**, **Github Actions** and **AWS**_.

Aditional information

- Languages:
  - Spanish (Native speaker)
  - English (Fully-conversational)

---

bayona.n@javeriana.edu.co • Bogota, Colombia
Reach me on **Telegram**, my username is nclsbayona
