CST8002 Programming Language Research Project

# Hybrid 04 – Build or Buy Library and/or API

**See Brightspace for Due Date**

**Refer to the Weekly Schedule document posted in Brightspace under Course Information for additional requirements common to all assessments.**

Microsoft Copilot [1] was used to generate the starter text for this Hybrid, which was then cross-checked using a web discussion on Quora [3]

## Overview

### What is a Library? What is an API?

* A library is a bundle of program functionality which is either created by a programmer for their own re-use across different projects, or a bundle that can be licensed for use from a 3rd party company(ies). Java programmers would use jar (**J**ava **Ar**chive) files, C# programmers would use dll (**d**ynamic **l**inked **l**ibraries) and so on. These would be incorporated into the project.
* An API (Application Programmer Interface) used to be synonymous with library, however API can also be used for program functionality that is not bundled within the project but provides services to the program from remote endpoints, microservice architecture, and integrated web services fall into this category. For example, mapping services, stock market services, weather services, movie showtime services, and so on.

### Why this Hybrid? (Tasks)

The focus of this hybrid is to develop an awareness of what are the things to consider when choosing between adopting a library/API or writing all your own code. This is important because in your practical project 3 there is a chance you will need to make this decision, and it is almost assured that you will need to make this decision as part of practical project 4.

* You will be asked quiz questions on parts 1 and 2.

## Part 1 – Build or Buy Library or API

When deciding between writing your own code, or adopting a library [1] suggests:

* that you will select writing your own code if you need customization not found in libraries, you want to learn more about coding, you want to optimize the code for your context, you do not want to have external dependencies.
* that you will adopt libraries is when you want to save development time, you want to adopt reliable and well tested functionality, there is community support, and letting you focus on priorities rather than ‘reinvent the wheel’.

When deciding between writing your own API, or adopting an API [2] suggests:

* that you will select writing your own code if you need customization not found in other APIs, you need full control over the API, you have need of customized security measures, you need to customize the scalability, you want to learn by building something unique.
* that you will adopt API(s) when adopting will save time and money, you want a reliable and tested API, want support and documentation, want to focus on core business activities, and want an API that has compliance with industry standards.

For both Libraries and APIs [1] and [2] also note that additional things you should consider are the licensing and costs (licensing costs versus your development and maintenance costs), the security of the API, how well the library or API is maintained, if the library or API is compatible with or integrates with your software, does the library or API perform to expectations, and will adopting either the Library or API lock you into a relationship with a vendor especially if your needs or the vendors licensing changes.

Something that neither [1] nor [2] discuss is the expertise of your team. The author of this document recommends that unless you have specialists / experts on staff for development areas like encryption and/or credit card transactions and/or banking transactions (and similar) that libraries or APIs created by experts be adopted due to regulatory compliance and guarantees.

The web forums by Quora [3][4] lists responses from industry practitioners on this topic as well and show that the AI generated information above is correct, but very general. You should review [3][4] briefly.

See the Appendix at the bottom of this document for the original text generated by [1] and [2].

## Part 2 – If buying library or API, review licensing.

For part 2 of this hybrid some sample library and API license documents are provided for your review, please review carefully as some hybrid quiz questions will come from the focused parts.

[6] and [7] are an example of a Database Connector Library, in practical project 3 some of you may decide to use a database and connect to it. The connector library usually is provided by the database vendor, and reviewing the license is important. Both [6] and [7] are for a Python connector to MySQL, however there is a commercial license and a community license depending on how the database itself is licensed.

[8] is an open-source license from the Free Software Foundation, it is used here as an example of open-source licensing but there are other open-source licensing agreements. Review these carefully, if your project and business model is closed-source, an open-source license for a library may force you to make your project into open-source if you adopt the library.

[9] is the license for the Google Maps API, this is an example of the license for a commercial web service. You would want to review these types of licenses for payment terms, as well as what you are permitted, and not permitted to do with the API.

[6] <https://downloads.mysql.com/docs/licenses/connector-python-9.1-com-en.pdf>

[7] <https://downloads.mysql.com/docs/licenses/connector-python-9.1-gpl-en.pdf>

These are lengthy documents (~67 pages long), focus on the table of contents for this hybrid.  
For [6][7] review and note the names of the licenses for third-party components.

For [6][7] Review and note the names of the standard licenses.

[8] <https://www.gnu.org/licenses/gpl-3.0.en.html>

For [8] scroll down to the bottom of the license and review the section “How to Apply These Terms to Your New Programs”.

[9] <https://cloud.google.com/maps-platform/terms>

For [9] is there a section on payment terms?

For [9] find the section under “3. License” and review what a customer is not permitted to do.

## Grading (10 Points Total)

* Hybrid Quiz 04 – 10 questions, 1 point per question.

# Sources Cited

Sources [1] and [2] follow the recommendations of the Algonquin College Library for citing AI tools, see [5], an APA Style was used for these as IEEE does not provide detailed style guidelines at the time of this writing.

Sources [3] and [4] are web forums with many participants, note that posts that constitute advertising or that are not relevant to the topic were omitted from the authors list here.

[1] Microsoft. (2024). Microsoft Copilot (November 2024 version) [GPT-4 Architecture, Large Language Model]. <https://copilot.cloud.microsoft/>

[2] Microsoft. (2024). Microsoft Copilot (November 2024 version) [GPT-4 Architecture, Large Language Model]. <https://copilot.cloud.microsoft/>

[3] Navin Kabra, Victoria J. Chin, Paul K. Young, Assistant Bot, Tony deAraujo, Sergey Ignatchenko, Lewis Pham, Stephen McConnell, Reinder de Vries, Zoran Bogicevic, Nikolay Konovalov, Diogo Neves, Lee Campbell, Vineet Shetty, David Kra, Jeff Benshetler, Len Lattanzi, Christian Ortmann, Ryan Cook, Tim Mensch, Christian Baune, Paul K. Young, Tom Nicholson, Don Yessick, Alejandro Radisic, Walt Smith, Ricardo Kagawa, Markus Mahlberg, Nguyễn Ái Quốc, Karoly Nyisztor, Michael Finney, Prateek Dilip Halwe, Hans Brinkhof, Edwin Vincent, Onimisi Onipe, Scott Deerwester,

Robby Goetschalckx, Tamara Roberson, Greg Kemnitz, Thirumal Venkat, Pawan Patel, Wim ten Brink, Og Lewisone, Jamsheed Kamardeen, Julie Haugh, Jan M Savage, Bojan Bedrač, Hayden Jones, Joe Wezorek, Jonah Miller, Kushagra Kumar, Matthew Parks Jr, Anurag Sharan, Laran Evans, Chip Frank, William Hurley, Aaron Christianson, Dale Strickler. (2024). Should I write my own code or use libraries?. quora.com. [online] Available at <https://www.quora.com/Should-I-write-my-own-code-or-use-libraries> [Accessed on Dec 18, 2024]

[4] Assistant Bot, Reinder de Vries, Karoly Nyisztor, Ankur Kulkarni, Edwin Vincent, Navin Kabra, Lewis Pham, Andrew Wiley, Paul K. Young, Sergey Ignatchenko, Your Lord and Savior Lucifer Morningstar, Tony deAraujo, Dale Strickler, Rodolfo Rodriguez, Tom Calver, Truly Mittal, James H. Kelly, Michael Cho, IT Guru USA, Avisake Mitra, Avisake Mitra, Aaron Rothschild, Len Lattanzi, James R. Twine, Richard Kenneth Eng, Igor De Carvalho, Saransh Bhandari, Apponward Technologies. (2024). How should I program my app? Should I use an API or write my own code?. quora.com. [online] Available at <https://www.quora.com/How-should-I-program-my-app-Should-I-use-an-API-or-write-my-own-code> [Accessed on Dec 18, 2024]

[5] Algonquin College. (2024). AI: Generative Artificial Intelligence: Citing AI Content by Citation Style. Algonquincollege.libguides.com. [online] Available at <https://algonquincollege.libguides.com/generativeai/citing> [Accessed on Dec 18, 2024]

[6] Oracle Corporation. (November 2024). Licensing Information User Manual MySQL Connector/Python 9.1.0 Commercial. downloads.mysql.com. [online] Available at <https://downloads.mysql.com/docs/licenses/connector-python-9.1-com-en.pdf> [Accessed on Dec 18, 2024]

[7] Oracle Corporation. (November 2024). Licensing Information User Manual MySQL Connector/Python 9.1.0 Community. downloads.mysql.com. [online] Available at <https://downloads.mysql.com/docs/licenses/connector-python-9.1-gpl-en.pdf> [Accessed on Dec 18, 2024]

[8] Free Software Foundation. (June 29, 2007). GNU General Public License, Version 3. www.gnu.org. [online] Available at <https://www.gnu.org/licenses/gpl-3.0.en.html> [Accessed on Dec 18, 2024]

[9] Google. (May 6, 2020). Google Maps Platform Terms of Service. cloud.google.com. [online] Available at <https://cloud.google.com/maps-platform/terms> [Accessed on Dec 18, 2024]

# Appendix: Microsoft Co-Pilot Chat History

Nov 30, 2024, for citation [1]

Prompt: What are the considerations for deciding between writing code from scratch, or using libraries?

Great question! Deciding whether to write code from scratch or use libraries involves several considerations:

**When to Write Code from Scratch:**

1. **Customization**: If you need highly specific functionality that existing libraries don't offer.
2. **Learning**: Writing from scratch can be a great way to deepen your understanding of a particular technology or algorithm.
3. **Performance**: Sometimes, custom code can be optimized better for your specific use case.
4. **Dependencies**: Reducing external dependencies can make your project more stable and easier to maintain.

**When to Use Libraries:**

1. **Efficiency**: Libraries can save a lot of development time by providing pre-built, tested, and optimized code.
2. **Reliability**: Established libraries are often well-tested and maintained, reducing the likelihood of bugs.
3. **Community Support**: Popular libraries usually have extensive documentation and community support, making it easier to find solutions to problems.
4. **Focus**: Using libraries allows you to focus on the unique aspects of your project rather than reinventing the wheel.

**Other Considerations:**

* **Licensing**: Ensure the library's license is compatible with your project's license.
* **Security**: Evaluate the security of the library, especially if it's handling sensitive data.
* **Maintenance**: Check if the library is actively maintained and updated.
* **Compatibility**: Ensure the library is compatible with your project's technology stack.

Ultimately, the decision depends on your project's requirements, your team's expertise, and the resources available. Do you have a specific project in mind where you're facing this decision?

Dec 2, 2024, for citation [2]

Prompt: What are the considerations for deciding between writing an API from scratch, or buying one?

Deciding whether to write an API from scratch or buy one involves several key considerations:

**When to Write an API from Scratch:**

1. **Customization**: If you need specific functionality or integration that off-the-shelf APIs don't provide.
2. **Control**: Full control over the API's design, implementation, and updates.
3. **Security**: Ability to implement custom security measures tailored to your needs.
4. **Scalability**: Custom APIs can be optimized for your specific performance and scalability requirements.
5. **Learning and Innovation**: Opportunity to innovate and learn by building something unique.

**When to Buy an API:**

1. **Time and Cost Efficiency**: Pre-built APIs can save significant development time and costs.
2. **Reliability**: Established APIs are often well-tested and maintained, reducing the risk of bugs and downtime.
3. **Support and Documentation**: Access to professional support and comprehensive documentation.
4. **Focus on Core Business**: Allows your team to focus on core business activities rather than API development.
5. **Compliance**: Many commercial APIs come with built-in compliance with industry standards and regulations.

**Other Considerations:**

* **Cost**: Compare the long-term costs of developing and maintaining a custom API versus the subscription or licensing fees for a commercial API.
* **Integration**: Ensure the API integrates well with your existing systems and technology stack.
* **Vendor Lock-in**: Consider the risks of becoming dependent on a third-party provider.
* **Updates and Maintenance**: Evaluate the frequency and quality of updates and maintenance provided by the API vendor.
* **Performance**: Assess whether the commercial API meets your performance requirements.

Ultimately, the decision depends on your specific needs, resources, and long-term goals. Do you have a particular project or use case in mind where you're facing this decision?