quantiphi
Solving What Matters

Legal Implications in the Use of Generative AI in Software Coding

TABLE OF CONTENTS

01

EXECUTIVE SUMMARY

Artificial Intelligence (AI) and Machine Learning (ML) are deeply impacting various areas of business and technology. This paper talks about the impact and implications of Generative AI on software coding, and consequently on technology services providers. Coding tools and processes are evolving significantly with the advent of Large Language Models (LLMs). Proprietary as well as open source LLMs are being leveraged by such tools to provide greater functionalities towards improved quality and productivity of the development process. While this is happening, it is important to carefully review the legal implications for a service provider to discuss the legality of use of such tools in terms of confidentiality, intellectual property ownership and infringement.

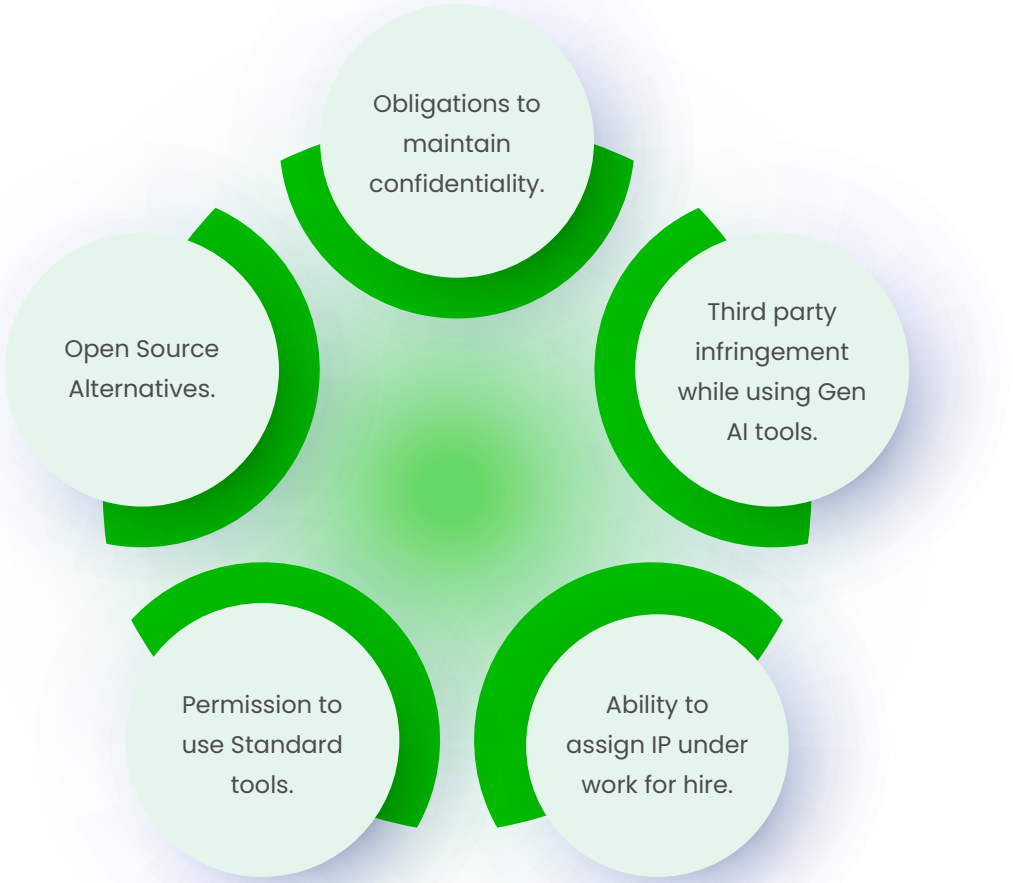
The paper analyzes terms and conditions of some of the most common generative AI based tools, on the touchstones of confidentiality, intellectual property ownership and infringement. The analysis results in the understanding that usage of generative AI tools based on certain legal parameters may be considered similar to that of non generative AI tools. Accordingly, protection of confidentiality and ownership of IP in the developed code largely remains the same as with non-generative AI tools. At the same time, some of the legal issues governing the use of Generative AI tools are yet to mature. Until such time that the pending issues are addressed and settled by adjudicating bodies, service providers should use internal policies and procedures to mitigate risks. Associated policies and procedures should be based on transparency, authorization, respect for intellectual property, awareness of bias, plagiarism, evaluation and, consent and attribution.

At the present stage we might not have answers to all the questions or situations that a service provider may encounter. However, by implementing robust policies and associated compliances, service providers should be able to minimize all kinds of risks associated with the use of Generative AI.

02

PROBLEM STATEMENT & RESEARCH

With the advent of generative AI (commonly referred to as GenAI) tools, the landscape of standard developmental tools is impacted for good. These tools, almost across the entire landscape, are evolving to include GenAI based features. These features have the potential to revolutionize the way code is written. At the same time, their usage is bound to impact legal aspects such as maintaining confidentiality, intellectual property ownership and infringement. Service providers are in a situation like never before, where generative AI tools bring in a strong use case, they also bring in complexities in how a service provider tackles the above mentioned legal issues.



The above issues are understood in this paper more elaborately in the backdrop of how the tools offer protection and comfort to service providers. Towards the end this paper tries to bring out recommendations for service providers for safe and ethical usage of Generative AI tools.

03

IMPACT OF GEN AI ON DEVELOPMENT TOOLS

Standard development tools are those that have widespread use in the software development community. Such tools are considered as permissible for use by the service providers as they develop applications for their enterprise customers.

Many such tools have been in use for decades now, leading to a good maturity in their functionality and understanding amongst large numbers of service providers globally. Let's see how GenAI is impacting tool categories:

Category of Tool	Description	Examples**	Gen AI Features*
Integrated Development Environment (IDE)	A comprehensive development environment with features like code editing, debugging, version control and project management	PyCharm or Jupyter Notebooks, VS code, Extensions such as Githib Copilot	Code completion, code generation, and bug detection
Data Visualization and BI Tools	Tools for preparing raw or unstructured data and making it acceptable for a machine learning model. Tools to provide relevant insights from data via interactive dashboards, reports, and analytics.	Power-2BI, Tableau, Looker	Automated and accelerated BI, Natural language interface for generating and modifying visualizations, summarizing reports and accelerated report generation
Testing Tools	Testing tools for code development are software applications that help service providers and testers to write better code and ensure its quality. These tools can be used to automate testing tasks, identify and fix defects, and measure code coverage.	CodiumAI, Applitools etc.	Test case generation, model based testing, debugging, unit-tests.

03

Category of Tool	Description	Examples	Gen AI Features*
Development and Version Control	Development and version control tools are software applications that help service providers to write, test, and manage code. Development tools provide developers with the environment and features they need to write code, such as code editors, compilers, and debuggers.	Github, Gitlab etc.	Model visualization and explanation.

**This is a non-exhaustive list and there may be other features related to Gen AI.tt*
***This is a non-exhaustive list.*

We can assess from the above table that generative AI has had a significant inroads with standard development tools. These tools now exhibit improved features relating to code refactoring, code generation, model visualization and bug detection to name a few. This trend is likely to continue, as Generative AI becomes more sophisticated and powerful.

03

In the following section, we have compiled terms and conditions of usage of hosted LLM tools. This paper utilizes the terms and conditions of widely available tools with the aim to create an understanding of various features that the terms have to offer with respect to confidentiality and IP ownership.

A simple overview of terms indicates that terms of each of the hosted tools have been updated to cater to the functionality of Generative AI. Terms such as opt outs and IP ownership specifically relate and describe gen AI specific terms which provide cushion to a service provider. More particularly, the terms are analyzed for enterprise versions of tools, based on following parameters:

OBLIGATIONS TO MAINTAIN CONFIDENTIALITY

When developing code for a client, it is important to take steps to protect the confidentiality of client's information. This is especially important when using third party tools, which may have access to the client's data.

Some steps that code developers generally take to maintain confidentiality across development engagements are as follows:

Encrypt the code:

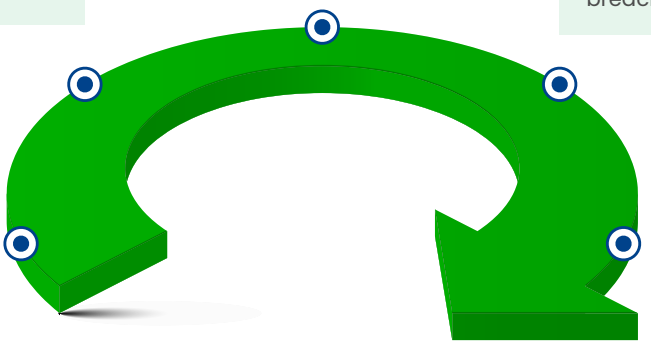
This will make it more difficult for unauthorized individuals to access the code.

Restrict access to the code:

Only authorized individuals should have access to the code, and access should be logged.

Monitor for security breaches:

Use security tools to monitor for potential security breaches.



Use a secure environment to develop the code:

This could include using a secure cloud environment or a virtual private network (VPN).

Have a plan for responding to security breaches:

If a security breach does occur, have a plan in place for how to respond.

03

Generative AI tools, such as Amazon Code Whisperer, Replit, Tabnine Open AI, and GitHub Copilot, use content, such as code snippets, comments, and content from files open in the IDE, to provide suggestions to users. This content is processed by the service solely to provide and maintain the service. As per the terms of use for enterprise, these tools may or may not retain client data for training.

Some of these tools may collect and use client-side telemetry and usage metrics for service improvement purposes. Most tools use client data for non-enterprise versions for training their models. In their enterprise versions, most of these generative AI tools provide options to not use their data for training purposes. Service providers have to be careful in selecting their tools on the basis of tier and options the tool provides for safeguarding confidentiality.

The difference between generative AI tools and non-generative AI tools is that generative AI tools send data to their large language models (LLMs) to receive a suggestion. Therefore, the data that was previously stored locally is now accessed by the LLMs for milliseconds to provide suggestions. This exposure of milliseconds has led to some hesitation in the industry, as it may be argued that this exposure leads to a breach in confidentiality. However, it is important to understand how these generative AI tools function. These tools have been pre-trained on large datasets and accordingly provide code suggestions. They provide suggestions based on the interrelation of the various sets of code lines. The underlying LLM model ascertains the interactions between the preceding code lines to predict what could be the best suitable next

set of lines. While doing so, the LLM model may provide multiple suggestions which would fit the technical character of the preceding code lines. It is quite possible that some of the suggestions provided by the tools may not pass muster and may not be relevant for a particular use case. So in each case, code developers would take the decision of accepting or rejecting a suggestion based on what suits best. The technical behavior of LLMs could be understood as that the actual code or confidentiality of the code is not hampered by these tools at any time. The entire code is sent as input to the LLM and the LLM then based on the inter-relation of code lines and on other parameters suggests the next set of lines. The concept of confidentiality in the realm of client and service provider, means that anything designated as confidential, should not be used in a manner other than what it is authorized for. In this unique case of gen AI tools, we can say that tools provide us with options where we can restrict the access of our codes solely to provide suggestions and the tools would not use our information to train their models. All these features are majorly made available in the enterprise versions. Once these features are enabled, confidentiality of the actual code is not hampered or affected by these tools. Accordingly, it could be said that though the functioning of gen AI tools may be considered different from non-gen AI tools, use of gen AI tools by service providers to deliver services may not affect the confidentiality of the code, if the service provider chooses the right tier and opt out options.

In conclusion, while there is some risk of confidentiality breaches associated with the use of any types of tools including generative AI tools, these risks can be mitigated in the same manner as they were being mitigated for non-gen AI tools. Service providers can choose LLM tools wisely and also take due care in selecting the right tier as well as opt out options. This risk is severely curtailed in cases where open source LLM models are used in designated environments. Having said that, a service provider should come up with best choices, safeguards in terms of policies and checks so as to have sufficient support for protecting confidential information. This paper provides information about suitable policies in the recommendations section.

03

THIRD PARTY IP INFRINGEMENT WHILE USING GEN AI TOOLS

Specifically, with regard to hosted Gen AI tools, most of the terms of such tools have been categorical that the ownership and responsibility of the lines of code suggested by the tool are that with the user. Let us take the case of GitHub Copilot, a hosted LLM product. The terms of use anoint the user of the tool as the creator and author of the code, however the matter is sub-judice, as GitHub is subject to litigation on the question of the ownership of the source materials of code that it has trained its models on.

It is noteworthy to mention that a class action lawsuit is pending adjudication before the US Federal Court of the State of California, on behalf of GitHub users against GitHub, Microsoft and Open AI. The contention being that GitHub Copilot has trained its model on public repositories that were made public under open source licenses that require attribution to the author. GitHub has denied this of course and claims that there is no specific code that is being copied. To its defense GitHub is using the argument of 'fair use' as the reason for not attributing the authors of the code, should the previous argument not fly. This case will majorly be argued on concepts of substantial similarity of copyright law.

The test of AI generated code being substantially similar to the code it was trained on is yet to be answered. Most of the tools state that the coders may use filters to check for duplication or similarity. Some of the tools have offered to indemnify if their suggested code infringes any third party's IP.

Accordingly, to an extent developers can utilize the inbuilt features along with the indemnification guarantee provided in the terms of use, to make themselves safe from any form of infringement. However, the judgment of the GitHub copilot case will remove much of the doubts that are currently prevailing over the use of generative AI tools.

In this phase of ambiguity and considering the strong use case of gen AI tools, service providers should focus on netting mechanisms such as open source software compliance review to pre-emptively catch substantial risks. This interim solution has been substantiated in the recommendation section of this paper

03

ABILITY TO ASSIGN IP DEVELOPED UNDER 'WORK FOR HIRE'

A work for hire contract is a legal agreement in which the creator of a work assigns all rights in the work to the hiring party. This means that the hiring party owns the copyright, trademark, and other intellectual property rights in the work.

In the context of software development, a work for hire contract is often used when a company hires a contractor to develop code for them. The work for hire contract ensures that the company owns all rights in the code, including the right to sell, distribute, and modify the code.

Considering the current scenario of tools, and as per the terms of use of commonly used tools all generative AI tools assign the IP in the code to the developer. Meaning thereby that the service provider owns the developed code. Depending on the contractual obligations with the client, the service provider can transfer the ownership of the code to the client.

All generative AI tools including products and open source both, provide the same chain of assignment as what was provided by non generative AI tools. On perusal of the terms of use of most of the available generative AI tools, we note that they possess the means to assign IP to the developer.

PERMISSION TO USE STANDARD DEVELOPMENT TOOLS

Standard tools have long been used in code development to improve productivity, efficiency, and quality. Generative AI tools are simply a new type of standard tool. They do not change the fundamental nature of code development, and they do not absolve service providers of their responsibility. Service providers still need to understand the problem they are trying to solve, come up with a solution, and then implement that solution in their code. They also need to test the code thoroughly to make sure that it is working correctly. The features disclosed in the above table clearly indicate that:

1. The codes generated by these tools are to be considered the intellectual property of the creator. The tools are simply providing suggestions, and the user is still responsible for the entire code. The user is also responsible for orchestrating the code together, including managing repositories and stitching the entire code as one piece and most importantly testing the code. Gen AI developed code will have to be tested thoroughly before it gets implemented, as the code suggested by gen AI tools may be marred by

bugs and may give unoptimized results. The code is not original to the tool, and it is not protected by copyright law.

2. The tools in their enterprise versions provide the user with options to select the right tier and opt out options so that the tool does not collect or store any confidential information or personally identifiable information (PII).

3. The tools are bound by their respective terms of use, which typically include provisions that protect the privacy of users and their data. These terms of use typically state that the tools will not collect or store any personally identifiable information without the user's consent.

While most of the above points have already been described in detail above, an overview of these points signifies that the tool providers in case of enterprise versions, have largely provided options for its users to keep their information safe and secure even after incorporating gen AI features. As a matter of fact, any service provider who chooses to use any AI development tool has to take care of the intellectual property developed, the confidential obligations and that the developed code does not infringe any third party intellectual property. With regard to the permissions for using these tools, we note that it is an understood fact that all service providers use various types of tools to develop codes for clients. In occasions where there is no contractual obligation of tools disclosure, service providers do not seek permission of the client to use such tools. Albeit, in situations where the service provider requires usage of tools which might affect client confidentiality, intellectual property ownership or would be against any policy of the client, service providers would need client's permission for using such tools. In cases where the service provider uses tools which provide safeguards such as opt outs and tier protection for enterprise versions, service providers do not need specific permission from the client, as it is common practice in the information technology industry that the service provider will use some tool to develop code, provided that the confidentiality and IP ownership is not affected.

Furthermore, with the advent of open source tools which can be used locally, service providers have the option of using tools which are not hosted over the internet. This local hosting of LLM will enable the service providers to easily transfer ownership of the generated code to its clients, in a manner similar to using non-gen AI tools.

03

Based on the analysis above we note that the nature of use of generative AI tools is largely similar to that of the existing tools. Though generative AI, when incorporated in the tools, brings in the complexity of using large language models, the usage shall remain the same as any standard tool. These tools are already pre-trained with terabytes of data and therefore result in improving the overall code development.

As with any standard tool, permission may be required to use generative AI tools in cases where the use affects the client's confidentiality and right to own IP. For example, if a tool exposes code to third parties, permission may be required to use such a tool. However, in general and based on the above table, permission to use generative AI tools should flow in the same way as permission to use other standard tools, i.e. where the code is exposed and confidentiality and intellectual property rights are affected, permission should be sought from the client. In all normal scenarios, where gen AI tools maintain confidentiality and provide ownership of IP, no permission may be sought.

The use of generative AI in standard development tools is clearly beneficial, as it helps save time and effort, improves the quality of code, and makes it easier to develop and deploy AI-powered applications to name a few. As a result, it is no longer a question of whether a tool provider should use generative AI. Rather, it is a question of how to best incorporate generative AI into tools to provide the best possible tool for developers. To conclude, standard development tools can no longer be separated from the influence of GenAI.

Accordingly, the research in this paper provides a clear pathway for the service providers to use gen AI tools for code development. However, as we wait for all the parameters to factor in, and as we wait for clarity on some of the legal issues, service providers may implement policies and procedures to curb the risk associated with these tools. While doing so, service providers also need to ensure that clients are assured to the maximum extent, and that they should have some form of transparency.

04

OPEN SOURCE ALTERNATIVES

By leveraging open-source LLMs, developers and service providers can circumvent the limitations of traditional LLM APIs and paid services, gaining access to a range of distinct advantages that align with the evolving needs of the modern technological landscape. These models empower service providers with greater control and adaptability, enabling them to customize and fine-tune the models to suit their specific requirements. Additionally, open-sourced LLMs eliminate vendor lock-in, allowing for seamless integration into diverse platforms without the constraints imposed by a single provider. Open-source models provide several benefits that make them highly desirable for programming and coding tasks. These benefits include:

1. **Flexibility and Customization:** Open-source models allow service providers to have full control over the model architecture, parameters, and code. This flexibility enables customization based on specific application requirements and helps service providers fine-tune the model for enhanced performance.
2. **Transparency and Code Review:** Open-source models provide complete access to the underlying source code, allowing service providers to review, understand, and audit the code. This transparency ensures trust and enables the identification and mitigation of potential vulnerabilities or biases.
3. **Reduced Dependence on Third-party Providers:** By relying on open-source models, service providers are not tied to a single provider or service. They have the freedom to use the model independently without relying on proprietary APIs or paying for commercial services. This reduces dependence and vendor lock-in risks.
4. **Access to a Vibrant Community:** Open-source models attract a large and active community of service providers who actively contribute to their improvement and maintenance. This community support fosters collaboration and enables service providers to learn from and share their experiences with peers, leading to continuous improvement and innovation.
5. **Cost Effectiveness:** Open-source models are generally available free of charge, reducing the financial burden for service providers and organizations. Additionally, open-source models can be deployed on local hardware or cloud infrastructure, providing cost-effective solutions compared to paid services.
6. **Enhanced Data Security and Privacy:** Open-source models allow service providers to process sensitive data locally, reducing the need to transfer data to third-party servers. This enhances data security and privacy, mitigating concerns related to data breaches or unauthorized access.
7. **Customized Development Environment:** Open-source models can be integrated seamlessly into a developer's preferred programming environment, enhancing

productivity and ease of use. service providers can leverage their existing tools and workflows, resulting in a more efficient development process.

8 **Latency:** Latency refers to the delay between the input and output of a system. Open-sourced models can offer lower latency as they can be run locally, reducing the need to communicate with external servers. This can result in faster response times, especially for real-time applications. On the other hand, large language model (LLM) APIs and paid services may introduce higher latency due to the need for external communication and processing, which can impact real-time performance.

9 **Predictability:** Open-sourced models can provide more predictability as developers have full visibility and control over the model's architecture, training data, and parameters. This allows for better understanding of the model's behavior and performance. In contrast, LLM APIs and paid services may offer less predictability as the inner workings of the models are often proprietary and not fully transparent to the users, which can make it challenging to anticipate their behavior in all scenarios.

10 **Ease-of-usage:** Open-sourced models can offer ease-of-usage through their flexibility and adaptability to specific use cases. Developers can customize and fine-tune these models to suit their needs. However, LLM APIs and paid services may provide a more user-friendly experience by offering ready-to-use solutions and seamless integration, which can be advantageous for developers seeking quick and convenient access to AI capabilities.

11 **Vendor Lock-in:** Open-sourced models generally avoid vendor lock-in, as they can be freely used, modified, and integrated into various platforms without being tied to a specific provider. In contrast, using LLM APIs and paid services may lead to vendor lock-in, as developers become dependent on the specific features, pricing, and terms of the service provider, potentially limiting their flexibility to switch to alternative solutions.

12 **Subject Matter Expertise & Manual-Resources:** Open-sourced models may require more subject matter expertise and manual resources for training, fine-tuning, and maintenance. Developers need to have the knowledge and skills to work with these models effectively. Conversely, LLM APIs and paid services can reduce the need for extensive expertise and manual resources, as they offer pre-trained models and automated services that require less hands-on intervention.

13 **Financial objectives such as Valuation (LLMs as assets):** From a financial perspective, open-sourced models can contribute to the valuation of a company or project by providing valuable intellectual property and assets that can be leveraged for various applications. These models can also enhance the overall technological capabilities and competitiveness of the organization. On the other hand, LLM APIs and paid services

may involve licensing fees, which can impact the financial objectives and valuation of a company. Additionally, the use of proprietary LLMs may limit the ownership and control of the underlying AI assets.

The benefits described have an operational impact and also provide long term solutions to issues discussed in this paper. As we decipher from the above mentioned pointers Open-source LLMs offer more flexibility and control over the code generation process. Service providers can fine-tune and customize the models to suit their specific needs, resulting in improved accuracy and performance. Moreover, Open-source LLMs provide opportunities for innovation, enabling service providers to push the boundaries of what is possible with LLMs. However, developing and maintaining Open-source LLMs require significant resources, expertise, and ongoing effort. Service providers need to invest in infrastructure, data collection, training, and model evaluation to ensure optimal performance over time. On the other hand, API based tools and LLM product offerings provide convenience and ease of use, especially for service providers who want to quickly integrate LLM capabilities into their applications or workflows. APIs abstract away the complexity of model development and maintenance, enabling service providers to leverage pre-trained models and additional features offered by the tool provider. API based tools and tool offerings also ensure compatibility across platforms and programming languages. However, using such tools may come with limitations in terms of customization and control. Service Providers are dependent on the model updates and availability, which may be subject to changes or limitations. Additionally, usage of such tools is also associated with costs, especially for high-volume or commercial use.

Open-source LLMs offer a compelling set of legal advantages that extend beyond their technical merits. Their inherent transparency and collaborative nature foster trust, facilitate knowledge sharing, and encourage the creation of more robust and reliable AI solutions. This open-source approach eliminates the need for complex licensing agreements with third-party vendors, reducing the risk of contractual disputes and ensuring companies retain control over their data and intellectual property. Additionally, open-source LLMs minimize the risk of patent infringement claims or copyright lawsuits, as the code is readily available for inspection and modification.

As organizations evaluate their specific needs, resources, and priorities, they must carefully consider the implications of choosing between custom models and APIs. Custom models offer greater control, customization, and long-term adaptability, while APIs provide convenience, rapid integration, and cost-effectiveness. Accordingly, open-source LLMs present a compelling alternative to traditional LLM APIs and paid services, providing a range of distinct advantages that align with the evolving needs of service providers and organizations. These models empower greater control, adaptability, and predictability for service providers while contributing to the valuation of companies and projects. As the AI landscape continues to evolve, open-source LLMs are poised to play an increasingly transformative role in shaping the future of artificial intelligence.

05

RECOMMENDATION FOR SERVICE PROVIDERS

The ongoing flood of generative AI tools and their sheer scale of their applicability has created an environment where adoption of AI is no longer a question. It is clear that we are at the cusp of an era where AI will permeate all cognitive and knowledge based work. The task is to steer and navigate it so that it aids and improves human effort, without marginalizing the human creators. Based on the analyses on the issues of confidentiality and ownership of IP in hosted LLM tools and open source LLM, discussed in this paper, it is noteworthy to mention that service providers depending on the choice of LLM tools, may or may not need additional permissions to use generative AI tools. Permissions sought for use of general standard developmental tools may be considered sufficient for use of gen AI based tools, where client confidentiality is protected. This includes the tools using enterprise versions that have an additional layer of protection for confidentiality. Having said the above, as a matter of abundant caution, service providers are advised to seek explicit permission from clients when using these LLM based tools.

While there are multiple issues related to hosted LLM tools, recently with the advent of open source tools these issues are almost non-existent. Open source tools can be used in any environment as the service provider or client may desire. Though these open source tools may feel less efficient or workable, their utility and ease of use trumps hosted LLM tools.

It is also important that some of the legal concepts with regard to usage of LLM tools are yet to be clearly laid down by courts and governments. In the absence of clarity on these legal concepts, it is of utmost importance that service providers take steps to mitigate the risks associated with these tools. While some of the risks could be mitigated with existing policies and procedures, for most part service providers will have to come up with additional protections in the form of choosing the right kind of tools, implementing the right kind of policies, providing attribution wherever applicable, and most importantly comforting its customers. Accordingly, the below points cover aspects for a Service Provider to take into consideration while using gen AI tools.

CHOOSE WISELY

As the field of generative AI grows with time, we will see the advent of more and more tools which are based on generative AI features. It will become imperative for a service provider to choose wisely from a pack of tools so that it enjoys the best of security, IP protection and privacy. When integrating large language models (LLMs) into service offerings, service providers must make a decision i.e. open-source or LLM-based product tools. Both options offer distinct advantages and present nuanced trade-offs, necessitating a data-driven approach to selecting the optimal solution. Some of the decision making aspects for choosing either open source LLM models or LLM based products Considering specific needs and priorities:

05

1. **Customization and Control:**

Open-source: Offers granular control over the model's architecture, training data, and biases, enabling bespoke solutions for unique tasks and data formats. This flexibility demands dedicated technical expertise and resources.

LLM-based tools: Provide readily available functionalities with limited customization options. While this expedites deployment, it restricts tailoring to specific needs.

2. **Technical Expertise and Resources:**

Open-source: Requires substantial AI expertise for setup, training, and maintenance, potentially creating a barrier for less tech-savvy service providers.

LLM-based tools: Offer user-friendly interfaces and minimal technical overhead, reducing reliance on internal AI expertise. This facilitates smoother integration for resource-constrained teams.

3. **Budgetary Constraints:**

Open-source: Generally cost-effective, with minimal licensing fees or ongoing subscriptions. However, additional costs may arise from hardware, infrastructure, and training resources.

LLM-based tools: Often involve licensing fees and subscriptions, potentially leading to higher total costs. However, these can be offset by reduced development time and technical expertise requirements.

4. **Performance and Feature Requirements:**

Open-source: May offer less fine-tuned performance compared to specialized LLM tools for specific tasks. However, open-source models can be optimized for specific tasks through custom training.

LLM-based tools: Often focus on specific industries or tasks and offer superior performance for those scenarios. However, they may lack the flexibility of open-source models for general tasks or unique needs.

Finding the Optimal Balance:

The ideal LLM solution lies in balancing control, customization, and ease of use within your specific context. This entails meticulously assessing your needs, resources, and budget constraints. Consider hybrid approaches utilizing open-source models combined with pre-built tools for specific functionalities. Conduct pilot tests with both options using actual data and tasks to objectively evaluate their suitability. Seek guidance from industry experts and consultants to navigate the complex LLM landscape and optimize your integration strategy.

05

By making a well-informed decision based on a comprehensive understanding of service needs and available LLM solutions, service providers can unlock the transformative potential of LLMs and drive growth from service offerings. While choosing within LLM based product tools, service providers choice of tools may be based on the following:

1. **The terms of service:** It is important to be comfortable with terms such as data retention, tenancy, code usage for training etc. Service providers need to ensure separate tenancy, no code retention, and transfer of ownership of IP.
2. **The security features:** Service providers need to ensure that the tool should have strong security features to protect the code from unauthorized access. This includes features such as encryption, access control, and auditing.
3. **The privacy policy:** The privacy policy of the tool should explain how the company will collect and use personal data. It is important to be comfortable with the privacy policy before using the tool.

By carefully considering these factors along with their sub factors such as code usage, tenancy, data encryption, data retention, a service provider can choose a generative AI tool that will help them protect their confidential information and IP, while also providing them with the technology they need to be successful. As the field of generative AI matures, it is important for service providers to be aware of the risks and challenges associated with using these tools. By choosing wisely and taking the necessary precautions, service providers can mitigate these risks and ensure that they are using generative AI tools in a safe and secure manner.

ATTRIBUTE

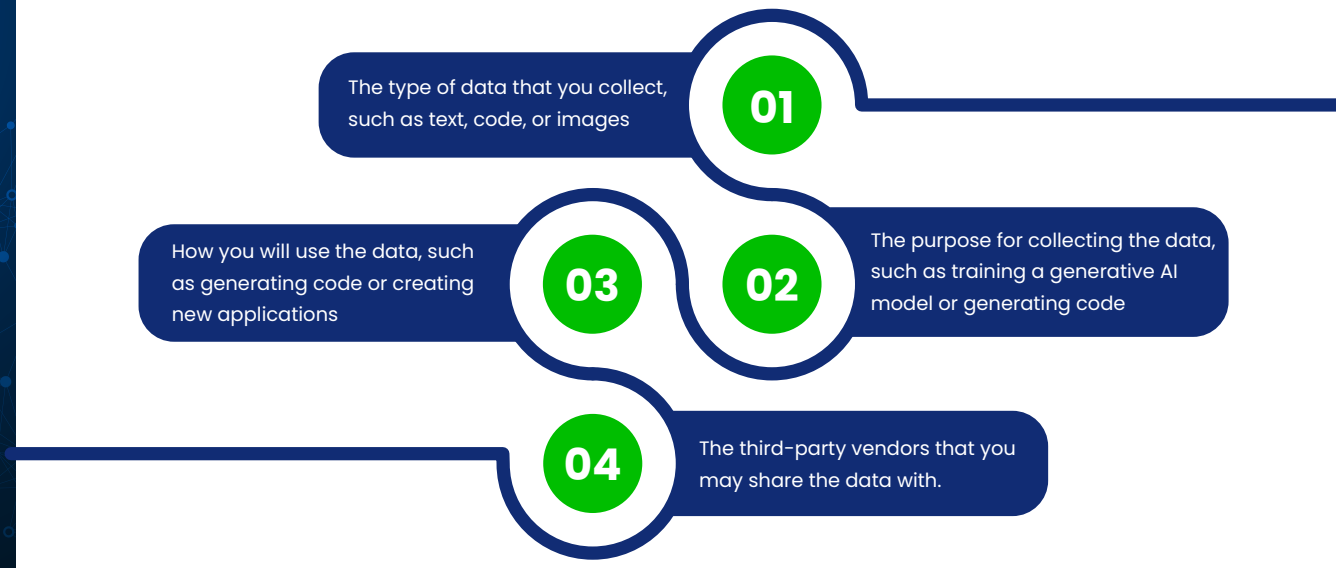
The AI model underlying a standard developmental tool may be trained on source code available in public code repositories, for example, open-source licensed code available on GitHub. The AI model becomes more and more efficient and accurate as it gets trained on these code sets. There are many intellectual property rights and ownership issues which are yet to be resolved. Accordingly, for Service Providers to not get entangled in these issues or atleast to be able to minimize liability and risks and in an ideal scenario it is important that service providers make plans to attribute the generated code to such licenses or code owners which the respective tools provides to them. This approach will create a balance between service providers and IP owners i.e. the coders/content creators whose code is used as a base by generative AI tools, by devising a system that allows for attribution and credit to the IP owners. Governments across the globe are trying to create AI governance frameworks which would include aspects of attribution.

05

However, we also note that at present providing attribution may only happen once the tool providers come up with a way to attribute. None of the tool providers currently take the path of providing attribution in their code suggestions. Accordingly, service providers should be prepared so that once the tool providers start giving the option to attribute, they should also pass the credit to the original coder.

ASSURE YOUR CUSTOMERS

A service provider should be transparent about its generative AI tool usage practices. It should let customers/clients know what data you (and the LLM tools) collect, why you collect it, and how you use it. This information should be easily accessible and easy to understand. This includes disclosing the following:



1. **Prepare policies for the usage of generative AI technology:** These policies should outline the ethical and responsible use of generative AI tools. They should also include procedures for handling potential risks, such as bias and privacy violations.
2. **Use client data responsibly:** Collect the data that you need and use it only for the purposes that you have disclosed to customers. This means avoiding collecting excessive data or using data for purposes that are not disclosed.
3. **Protect data security:** Take steps to protect the security of the data that you collect, such as using encryption and access controls. This will help to prevent unauthorized access or disclosure of data.

05

COMPLIANCE PROCEDURES

While the use of Generative AI may not warrant seeking additional permissions from the Client regarding use of such tools for providing services or deliverables, it is of utmost importance that we choose the right kind of tools, implement the right kind of policies, provide attribution wherever applicable, and most importantly comfort our customers. By creating and following compliance procedures based on the below concepts, service providers would be able to use generative AI tools responsibly for their clients as well as for their internal development:

1. **Transparency:** Service providers should be transparent with their team and clients wherever applicable, about their use of generative AI tools. Use of any open source tools may be exempted from this
2. **Authorization:** Service providers should make a graded mechanism where developers can use different types of LLMs based on their risk analysis, and wherever applicable seem authorizations within the org and/or from the client.
3. **Respect for Intellectual Property:** Developers of service providers should be respectful of the intellectual property rights of others. They should not use generative AI tools to generate code that infringes on the copyrights of others.
4. **Awareness of Bias:** Developers should be aware of the potential for bias in generative AI models, these may include automation bias, bias based on existing dataset, among other types of bias. They should be sure to review the code generated by generative AI tools for bias and to make any necessary corrections.

05

5 **Plagiarism Evaluation:** Service providers should endeavor to use tools which perform OSS review once the code is complete. This step will help in netting at least OSS licensed codes, resulting in further minimization of risk of using infringing code.

6 **Consent & Attribution:** Service providers may have mechanisms in place to seek consent from OSS licensed code owners. Where possible, Service providers should endeavor to provide attribution to the fullest extent possible.

By following these set of considerations, service providers can use generative AI tools in a professional and responsible manner in a work for hire context. While choosing the best possible generative AI tool for each use case, a service provider can very well utilize the various security features available in each tool, or they may use Open source tools to its benefit and be free from the legal issues inherently present in API based tools.