Department for Science, Innovation & Technology

Government Digital Service

AI Insights

AI Coding Assistants for developers in the UK Government

Table of Contents

Introduction

Artificial Intelligence coding assistants (AICAs) have existed for longer than the current era of generative AI. For example, GitHub's Copilot, which is powered by the code-specialised OpenAI Codex model, was introduced in July 2021. Compare this to the latest generative AI models which were mostly launched throughout 2023 and 2024.

As a result of its longer legacy, the market for AICAs has become increasingly diversified. All hyperscalers, and a large number of software and low-code industry leaders, offer AICA based on a variety of models. In addition, the open source community is beginning to extend the market. An example of this is NVIDIA's recent contribution to the open source StarCoder2 model of the BigCode community.

While quantitative evaluation of the impact of this technology is still ongoing, studies suggest that developers view AICA technology mostly positively. Furthermore, according to the 2023 Stack Overflow Developer Survey, 80% of respondents were already using the technology in private projects.

For senior leaders in His Majesty's Government (HMG), prohibiting or delaying the onboarding of AICA tools within their organisations pose three significant concerns:

1. the risk of falling behind in developer productivity
2. reduced developer experience and happiness
3. the unregulated use of AICA leading to unsanctioned shadow IT, with its associated risks

This guidance uses the principles of the AI Playbook for the UK Government to frame recommendations for the use of AI coding assistants.

The aim of these guidelines is to allow you to enjoy the benefits of AICAs while minimising the risks associated with unregulated use of this evolving technology.

This guidance has been developed in collaboration with senior stakeholders, technologists, and assurance teams to ensure that it covers the full strata of industry expertise, and can be used by individuals and organisations in both the public and the private sector.

Guidance

The 10 principles in the AI playbook for the UK Government have been designed to guide the use of artificial intelligence in government organisations. Each principle sets out the acceptable criteria for key areas. For example, to ensure that you can understand and account for the limitations of AI.

In this guidance, we have applied these principles more specifically to AICAs and their use cases.

Principles

Principle 1: You know what AI is and what its limitations are

Code completion and chat functionality

In the context of AI coding assistants, two types of AI model are primarily in use at the time of writing. These are:

1. computer code trained models, such as the Codex model of OpenAI which underpins GitHub Copilot, or the open source BigCode model StarCoder2
2. large language models (LLMs), or foundation models, such as the open source Llama or closed source GPT-4

These models are typically used for almost-real time code completion and their conversational capability, which can assist by processing complex instruction prompts and providing code explanations.

Providing context and data is important

Providing more context to AI models can dramatically improve their performance. The more specific the provided information - or prompt - is to the task, the better the model will perform.

The inability to produce original content

All AI models commonly in use in AICAs are in effect sophisticated text predictors. They do not have inherent logical capabilities due to being built on strictly deterministic computer systems.

AI as a highly specialised colleague

It is important to treat AI coding assistants as if they are a programmer who lacks experience and the ability to see the big picture in terms of business logic and the overall algorithm. This AI colleague, however, does have a perfect recall of programming languages and exceptional lookup skills.

Principle 2: You use AI lawfully, ethically and responsibly

The programmer is responsible for code produced

As is true for other code assistant tools and resources such as IntelliSense or Stack Overflow, the responsible use of AI coding assistants require you to understand the resulting code and take sole responsibility for any changes.

AICAs produce deterministic and testable output

AICAs produce code against easily testable, verifiable instructions so they are easier to safeguard against hallucinations and evaluate than that of other generative AI applications, such as those that produce free-form text.

Coding use-case limits ethical risks

Conversational-type coding assistant components are usually built on state of the art LLMs, so they will be subject to certain conditions in regards to decision making and testing for bias. If they are used only for explaining and generating code, the likelihood of controversial use is very low.

However, AICAs will not actively recognise or flag bias, so it remains up to you to make sure your code does not contain bias.

Intellectual property (IP), licensing, and indemnification

The most often discussed ethical and legal topics with regards to AI coding assistants are related to intellectual property rights and licensing issues. Licence violations will mainly take the form of proprietary codes being inadvertently introduced. Licence poisoning is also possible, which is when the usage of one licence, such as GPL3, requires the rest of the code to become GPL3 licensed as well.

Most enterprise or professional level AICA offerings include IP indemnity clauses that protect users from legal challenges for inadvertent use of proprietary code. Technical features can also be enabled to block the coding assistant from suggesting proprietary code. Depending on the AICA supplier's approach, they either assure that code suggestions are licence-free and non-proprietary, or they will provide you with the opportunity to see the source of the code. From this, you will be able to make an informed decision on usage.

Good practice reduces risks

The closer a development platform and deployment infrastructure is to good practice, the less concern you should have about the specific use of AI coding assistants.

You can greatly reduce the risks of employing AI coding assistants in your development environment by:

- working in the open
- employing main branch protections
- Maintaining the strict separation and audit of production secrets access
- using multi-stage deployment
- including sufficient test coverage and vulnerability scanning for continuous deployment in your pipelines

On the other hand, if your production service is developed, maintained and deployed from a single environment, using AI coding assistants may introduce unacceptable risks. An example of this would be of the AICA regularly communicating production secrets with its provider.

Only use trusted products

Only use integrated development environments (IDEs) and AI coding assistants provided by trusted vendors. The plugin infrastructure of many integrated development environments (IDEs) contain a large number of offerings of GPT-style assistants of undisclosed or unvetted origin. The use of such plugins will carry a heightened risk of data exposure and malicious code injection.

For more information, see principle 8 and principle 10.

Principle 3: You know how to use AI securely

Make sure you understand the terms and conditions of the software you are using

At this time, most proprietary AICAs that are available free of charge or on a personal licence level will record prompts and responses in order to train their provider's models. Due to this, it is strongly recommended that you only make use of business or enterprise level contracts, or local model execution of open source models (for more information, see principle 6).

If you are using the service of a trusted provider who stores private code, such as GitHub, the added risk should be acceptable to a business contract.
If in doubt, your information assurance (IA) colleagues or senior responsible owners can help to determine the feasibility of use for different use cases, including those involving private repositories. For more information, see principle 8 and principle 10.

You will also need to make sure that anything offered as beta or experimental features are not covered by different terms that may allow for data collection.

If you're in any doubt, you should coordinate with commercial colleagues in your organisation and use AICAs in alignment with your organisation's information assurance (IA) guidelines.

Make sure secrets are separated from AICAs

AI coding assistants work better the more context you provide to them. While an IDE plugin may provide all data open or present in a workspace to the AICA backend, you will need to be aware that any secrets present in your workspace will potentially be uploaded to the reasoning (or inference) layer.

You should never handle sensitive production data or secrets in development environments.

Direct access and deployment to production carries unacceptable risk

As a risk owner, you need to make sure that your production environment is insulated from development environment changes. The main branch and resulting artefacts need to be protected from unchecked modifications.

Keep software versions up to date and read the changelog

Make sure that you monitor versions for all AICA software and tooling. This includes plugin and model versions. Keep any local or remote software under your control updated to avoid known bugs and vulnerabilities. Also,make sure to read changelogs in order to check for breaking changes and licence or privilege changes

Use additional vulnerability scanning software

AICA producers advertise their awareness of vulnerabilities and their capacity to mitigate most of these automatically. Therefore, it is important to establish a multi-layered approach to vulnerability scanning.

In traditional software development best practice, additional third party tools such as Snyk Code or Aikido, or language-specific static analysis tools like Chekov or Tfsec, should be used to complement the use of AICAs.

Be wary of dependencies introduced by AI coding assistants

Several reports have noted that language models may hallucinate version numbers of library dependencies. If malicious actors become aware of these, they can provide made-up dependency versions that include malicious code.

Always double-check the version requirements of dependencies introduced by a coding assistant against trusted sources.

You must also make sure that you have meaningful human control (see principle 4).

Principle 4: You have meaningful human control at the right stages

Only accept code changes you understand

You should only commit code changes that you understand. As a developer, you are fully responsible for the code you produce, whether this is AI assisted or not.

Require peer reviews

Merges made to the main branch need to be subject to human peer review by one or more peers, and adhere to your organisation's policies.

It's best to use human peer review in combination with branch protection on your main branch.

Small and numerous commits

Keeping change volume per commit low and specific is good practice in order to better track changes and reduce overload for your reviewers. This extends to avoiding multi-commit merge requests where possible.

Principle 5: You understand how to manage the full AI lifecycle

Lifecycle management is plugin version management

In general, with integrated development environment (IDE) plugins, you will be able to control the version installed into your development system. It is good practice to keep these updated to avoid known bugs and vulnerabilities.

Make sure to read the changelog in order to check for breaking changes, and any licence or privilege changes. If applicable, also consider your organisation's update strategy.

Do not rely on non-deterministic outputs

Due to the non-deterministic nature of the models underpinning AI coding assistants, your code and build pipeline should never rely on a specific response to a prompt unless you are willing to test these responses extensively and accept the risk of frequent breakage.

Do not introduce dependencies on AICAs

The appropriate use of AI coding assistants should introduce no end product, code or documentation dependencies on the employed tool.

Used correctly, coding assistants can be interchanged flexibly by activating or deactivating plugins as required.

Experiment

Experimentation is strongly encouraged as, depending on your language and use case, AI coding assistants will display varying strengths and weaknesses.

Principle 6: You use the right tool for the job

Use the simplest model to meet your needs

You should be aware that the complexity and size of the underlying model will have a significant impact on an AICA's processing speed and energy consumption. For example, while code-specific models might be seen by some as inferior to more state of the art foundation models, they can produce comparable results faster and at a fraction of the energy cost.

Experiment with different models to best meet your use case

Different models can meet different language and use case needs. Experimentation is encouraged, especially when working with legacy (or mainframe) specific languages such as COBOL or Natural.

Consider open source

The Service Standard and the Technology Code of Practice (TCoP) support using and contributing to open standards, common components and patterns.

Open source models with industry backing such as StarCoder2 or Gemma2, and tooling such as Ollama and Continue, are increasingly performant and easy to use. These may suit use cases where local model execution, with comparably high machine performance requirements and operational overhead, are feasible.

Principle 7: You are open and collaborative

Consider open source first

In keeping with the Service Standard and the Technology Code of Practice (TCoP), we strongly recommend coding in the open. You should also follow open source guides to ensure best practice.

Give experts space and time to share

To share experiences with new technology and good practice, allow yourself time to document, share and discuss your learnings, and listen to others. AICA champions can greatly enhance organisational learning and productivity in new technology.

Share knowledge and experience in your organisation

Consider scheduling regular knowledge sharing sessions and expert surgeries to enhance learnings and avoid duplicated efforts.

Share knowledge and experience with the broader community

Celebrate learnings and success stories by sharing them with the cross governmental community under the Cross Government Software Engineering Community.

Principle 8: You work with commercial colleagues from the start

Coordinate with commercial colleagues in your organisation

Commercial colleagues will have knowledge of commercial processes and requirements such as organisational policy compliance with processes such as commercial and technical information assurance (IA) and data protection impact assessments (DPIA).

Make sure your use case and AICA are compliant with your organisation's policies (see also principle 10 below).

Principle 9: You have the skills and expertise needed to implement and use AI solutions

Check your contracts for learning resources

Many contracts with strategic partners such as Microsoft, IBM, Google or AWS include learning credits which may include access to resources and training on AICAs.

If you are unsure, consider consulting with your commercial team to find out what might be available.

Make sure to include training in your onboarding process

Qualitative studies suggest that, as with all new technology, training significantly enhances the experience and productivity when using AICAs.

Training credits may be available to you through your organisation or existing contracts.

In central government, further support may be available. For more information, contact the CDDO Engineering Excellence Team.

Identify experts in your organisation

Products such as GitHub Copilot and IntelliSense have been available for years, so your colleagues may be able to provide expert knowledge. Learning from experienced users can greatly contribute to organisational learning.

Principle 10: You use these principles alongside your organisation's policies and have the right assurance in place

Use AICAs in alignment with your organisation's information assurance (IA) processes

Always reach out to your organisation's IA team to ensure that you're adhering to the correct processes. and processes.

Regularly check for updates of central guidance

The market for AI tooling, and AICAs, is developing quickly. Various work streams in the public sector are evaluating these tools and moving towards putting them into production use.

This guidance will be updated regularly to reflect new findings and policies so you may find it useful to check back regularly.