Jaguar

User Guide

Contents

1 Introduction

Welcome to the Jaguar User Guide! This comprehensive guide is designed to help you effectively utilize Jaguar, an integrated development environment (IDE) specifically tailored for Tiger and Java programming languages. Whether you are a beginner or an experienced programmer, this user guide will provide you with the necessary information to navigate Jaguar's features and maximize your productivity.

1.1 Who is Jaguar for

This user guide is intended for developers, programmers, and anyone who will be using Jaguar for Tiger and Java programming. Familiarity with these programming languages is beneficial, but not required, as this guide will provide explanations and examples to facilitate understanding.

To use Jaguar, you will need a compatible operating system and the necessary system requirements. Please refer to the installation instructions in the next section for more details on the prerequisites.

Now, let's dive in and get started with Jaguar!

2 Getting Started

Before you can start using the Jaguar, make sure you meet the following prerequisites and follow the installation instructions provided in this section.

2.1 Prerequisites

To use Jaguar, you need the following:

- Operating System: Windows 10, macOS 11, or Linux (Nixos)
- Minimum RAM: 4 GB
- Disk Space: 5 GB of free space
- Java Development Kit (JDK) 11 or later installed
- Maven

Ensure that your system meets these requirements before proceeding with the installation.

2.2 Installation Instructions

Follow these steps to install Jaguar on your system:

2.2.1 Windows

1. Download the Jaguar installer for Windows from the official website https://uss-roosevelt.github.io/jaguar_website/.

2. Double-click the installer file and follow the on-screen instructions.

3. Once the installation is complete, you can launch Jaguar from the Start menu or desktop shortcut.

2.2.2 macOS

1. Download the Jaguar app file for macOS from the official website https://uss-roosevelt.github.io/jaguar_website/.

2. Unzip the archive.

3. Launch the `.app` executable.

2.2.3 NixOS

1. Open a terminal and navigate to the directory where you want to install Jaguar.

2. Dowwload the NIXOS.TAR.GZ tarball using the following command

```
$ wget https://uss-roosevelt.github.io/jaguar_website/files/nixos-jaguar.tar.
```

3. Extract the files

```
$ tar -xvf nixos.tar.gz
```

4. Build the package.

```
$ nix-build
```

5. To acces the app

```
$ nix-shell -A pingEnv
```

6. In this shell the app is on your path

```
$ jaguar
```

3 Basic Concepts

Before diving into the details of using the Jaguar, it's important to familiarize yourself with some key concepts and terms that will be referenced throughout the user guide. Understanding these basic concepts will help you navigate and utilize the Jaguar more effectively.

3.1 Jaguar

Jaguar is an integrated development environment (IDE) designed specifically for Tiger and Java programming languages. It provides a feature-rich environment for writing, editing and executing code. Jaguar offers a comprehensive set of tools and features to streamline your development workflow and enhance productivity.

3.2 Tiger and Java Programming Languages

Jaguar supports two programming languages: Tiger and Java. Here's a brief overview of each language:

3.2.1 Tiger

The Tiger programming language is a statically-typed programming language that was created as part of a textbook called "Modern Compiler Implementation in ML" by Andrew W. Appel. The language was designed to serve as a teaching tool for compiler construction and programming language concepts.

Here are some key features and characteristics of the Tiger programming language:

- **Syntax:** Tiger has a C-like syntax, with familiar constructs like loops, conditionals, functions, and variable declarations.
- **Static Typing:** Tiger is statically typed, which means that variable types are checked at compile-time, helping to catch type-related errors before runtime.
- **Strong Typing:** Tiger enforces strong typing, meaning that type conversions and operations between incompatible types are not allowed by default.
- **Support for Basic Data Types:** Tiger supports basic data types such as integers, strings, arrays, and records (struct-like data structures).
- **Functions and Procedures:** Tiger allows the definition of functions and procedures. Functions can have return types, while procedures perform actions without returning a value.

- **Memory Management:** Tiger does not provide automatic memory management features like garbage collection. Memory allocation and deallocation are handled explicitly by the programmer.
- **Limited Standard Library:** Tiger has a minimalistic standard library, providing basic input/output operations, string manipulation, and simple mathematical functions.

It's important to note that the Tiger programming language was primarily created as an educational tool and is not widely used outside of the context of compiler construction courses or related academic purposes.

3.2.2 Java

Java is a widely-used, general-purpose, high-level programming language that was originally developed by Sun Microsystems (now owned by Oracle Corporation) in the mid-1990s. Known for its "write once, run anywhere" philosophy, Java allows developers to build applications that can run on various platforms without the need for recompilation.

Here are some key features and characteristics of the Java programming language:

- **Object-Oriented:** Java is primarily an object-oriented programming language, which means that it focuses on creating objects that encapsulate data and behavior.
- **Platform Independence:** Java programs are compiled into an intermediate bytecode format that can be executed on any platform with a Java Virtual Machine (JVM). This makes Java highly portable and platform-independent.
- **Garbage Collection:** Java provides automatic memory management through garbage collection. The JVM automatically manages memory allocation and deallocation, freeing developers from explicit memory management tasks.
- **Strong Typing** Java enforces strong typing, meaning that variable types are checked at compile-time to ensure type safety and prevent certain programming errors.
- **Rich Standard Library:** Java comes with a comprehensive standard library that provides a wide range of classes and APIs for performing common tasks, such as input/output operations, networking, database connectivity, and more.
- **Multithreading:** Java has built-in support for multithreading, allowing developers to create concurrent and parallel programs with ease.

- **Exception Handling:** Java has a robust exception handling mechanism that enables developers to handle and recover from runtime errors gracefully.

Java's versatility, portability, and extensive ecosystem have contributed to its popularity in a wide range of domains, including web development, mobile app development (using frameworks like Android), scientific computing, financial applications, and more.

3.3 Workspace and Project Structure

The Jaguar workspace is the top-level container for all your projects. It provides a central location where you can organize, manage, and work on multiple projects simultaneously. Within the workspace, each project has its own structure, consisting of files, folders, and dependencies.

3.3.1 Files and Folders

In Jaguar, you interact with source code files, resource files, and other project-related files. These files are organized into folders within the project structure. Understanding the layout and purpose of different folders will help you navigate and locate your project resources efficiently.

Now that you have a better understanding of the basic concepts related to Jaguar, let's move on to the next section to learn how to create and work with projects in the IDE.

4 Starting with Jaguar

Once Jaguar is installed, you will be able to run it. In this section, we are presenting you the information to be found on the starter page, as well as how to manage your projects within Jaguar.

4.1 Overview of the Starter Page

Once you launch Jaguar, you will head to the Starter Page. From here, you will be able to create new projects, open projects on your local storage and also access the settings page.

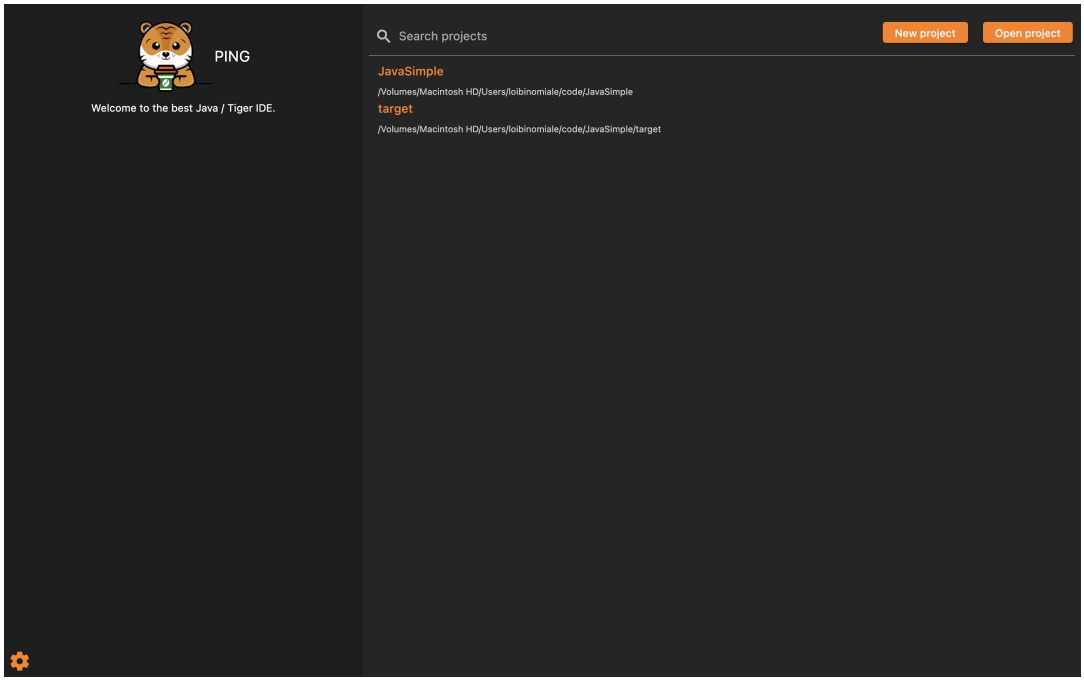

Figure 1: Welcome to Jaguar!

From this page, you can see the history of the last open projects in Jaguar, and you can also filter them by typing on the search bar in the top section. If you click on the project's name, you will head to the code editing page and be able to start coding.

4.2 Creating a New Project

To create a new project in Jaguar, follow these steps:

1. From the Starter Page or the Workspace, click on `New Project` in the top right corner.

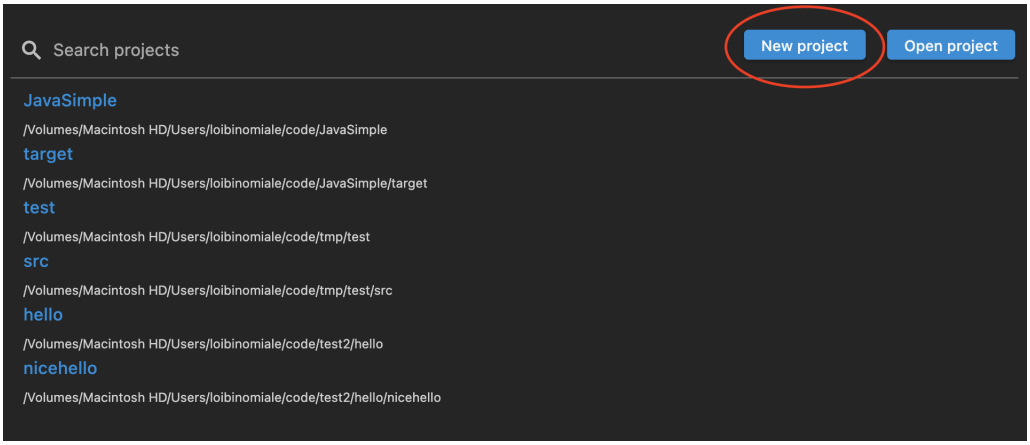

Figure 2: Creating a new project from the Starter Page

2. Once you are on the `New Project` page, decide whether you want to use a project template or just create an empty project by choosing in the top left corner.

3. Specify the project name, location, and any additional configuration settings as the project template.

4. Click `Create` to create the project. Jaguar will generate the necessary project structure and head you to the Workspace.

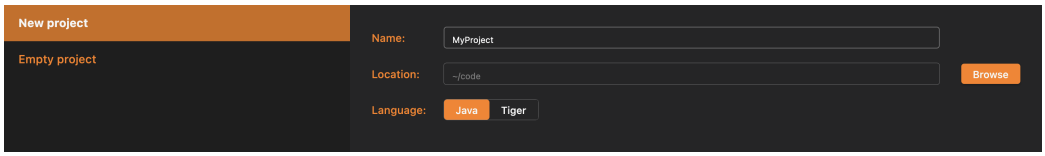

Figure 3: Creating our first project

4.3 Open Existing Projects

If you already have an existing project that you want to work on in Jaguar, you can open it using the following steps:

1. From the Starter Page or the Workspace, click on `Open Project`. It will open a file explorer.

2. Choose the root directory of your project.

3. Click `Choose` to import the project into Jaguar. The opened project will now appear in the workspace, ready for editing and execution.

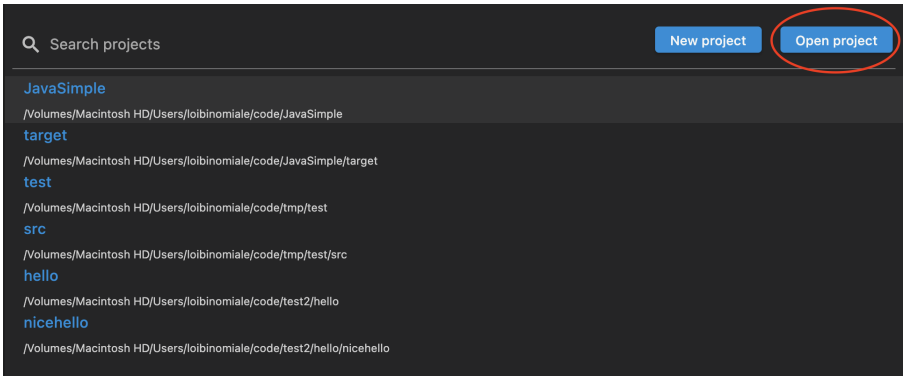

Figure 4: Opening a new project from the Starter Page

5 Editing Code

Editing code is a fundamental part of the development process, and Jaguar provides a range of features and tools to enhance your coding experience.

This section will introduce you to some of the key code editing capabilities available in Jaguar.

5.1 The Workspace

You will likely spend most of your time on Jaguar in the workspace, so let's take a look to it.

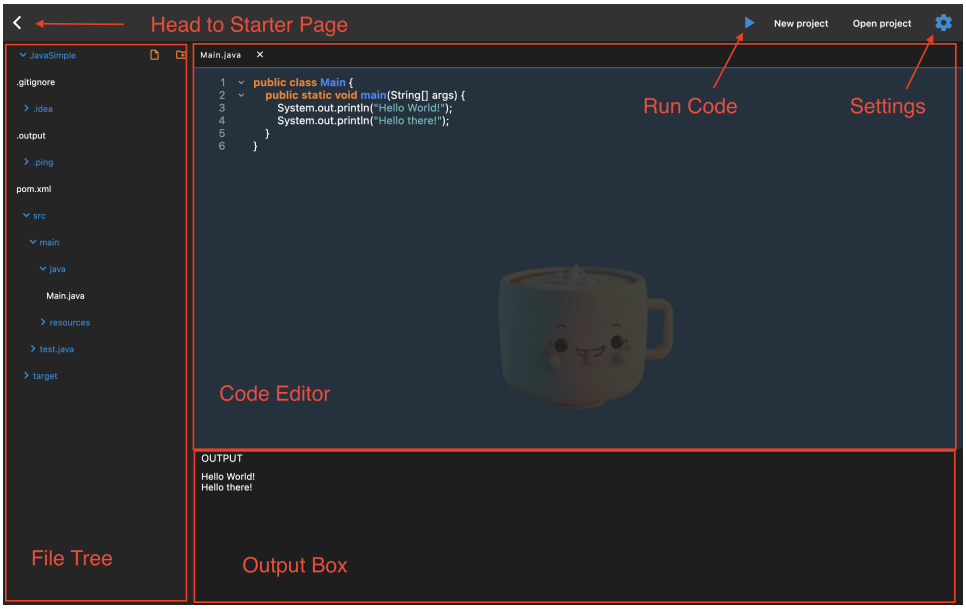

Figure 5: Opening a new project from the Starter Page

You might have noticed that you can go back to the `Starter Page` and that

you can go also go into the `Settings` page. We will dive into the settings later in this user guide.

5.2 Creating and Opening Code Files

To create a new code file in your project, follow these steps:

1. In the File Tree , click on the `New File` button.

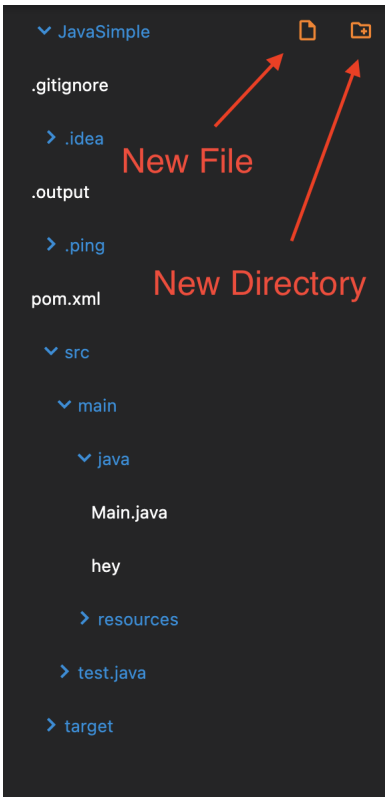

Figure 6: Opening a new project from the Starter Page

2. Enter the desired file name and select the file location.
3. Click on `Ok`.

To open an existing code file in Jaguar, simply click on the file in the File Tree.

Please note that you can have several files open at the same time, and select the file you want at the top by clicking on their name.

5.3 Editing Features

Jaguar offers a variety of editing features to improve your coding productivity. Here are some of the key features you can leverage:

5.3.1 Syntax Highlighting

Jaguar provides syntax highlighting, which visually distinguishes different elements of your code by applying different colors. This feature helps you quickly identify keywords, comments, variables, and other language-specific constructs.

5.3.2 Code Completion

Code completion is a powerful feature in Jaguar that suggests language-specific constructs, methods, and variable names as you type. It speeds up your coding process by reducing manual typing and helping you discover available options within your codebase.

6 Running Code

Once you have written your code in Jaguar, you can easily run and execute it to see the results. This section will guide you through the process of running your code within Jaguar.

6.1 Running Code in Jaguar

6.1.1 Running Java Code

To run your Java code in Jaguar, follow these steps:

1. From the top bar, click on the `Run` button.
2. Look at the output box at the bottom, it should show the output of your program. You can scroll up and down the output box to make it larger or smaller as you please.

6.1.2 Running Tiger Code

You can run your Tiger programs in Jaguar with a local compiler you provided or a remote compiler.

- **With a local Tiger compiler:**
  1. Set the Tiger Compiler to `Local` and select the path to your local Tiger compiler in the settings.
  2. Make sure the code file you want to run is open in the Jaguar editor.
  3. From the top bar, click on the `Run` button.
  4. Look at the output box at the bottom, it should show the output of your program. You can scroll up and down the output box to make it larger or smaller as you please.
- **With the remote Jaguar compiler:**
  1. Make sure you have an internet connection.
  2. Set the Tiger Compiler to `Remote` in the settings.
  3. Make sure the code file you want to run is open in the Jaguar editor.
  4. From the top bar, click on the `Run` button.
  5. Look at the output box at the bottom, it should show the output of your program. You can scroll up and down the output box to make it larger or smaller as you please.

6.2 Working with Program Arguments

If your code requires command-line arguments or program parameters, you can specify them when running your code in Jaguar. Follow these steps:

1. Open the code file in the Jaguar editor.

2. From the menu bar, select `Run > Run Configurations`.

3. In the "Run Configurations" dialog, locate the appropriate run configuration for your code.

4. Select the run configuration and navigate to the "Arguments" tab.

5. Enter the required program arguments in the provided field.

6. Click "Apply" or "Run" to execute your code with the specified program arguments.

6.3 Handling Output and Errors

The output of your code and any error messages will be displayed in the console view. It's essential to review the console output carefully to verify the correctness of your program and address any runtime errors or exceptions that may occur.

Make sure to utilize error handling mechanisms, such as try-catch blocks or exception handling, to handle and manage errors gracefully within your code.

Congratulations! You have successfully learned how to run your code in Jaguar. In the next section, we'll cover how to have full control over the IDE and use the settings to improve your experience.

7 Settings

In addition to the core functionalities mentioned earlier, Jaguar provides settings that let you have a better control over the IDE and its features.

You can access the `Settings` page from the bottom-left corner of the `Starter Page` or the top-right corner of the `Workspace`.

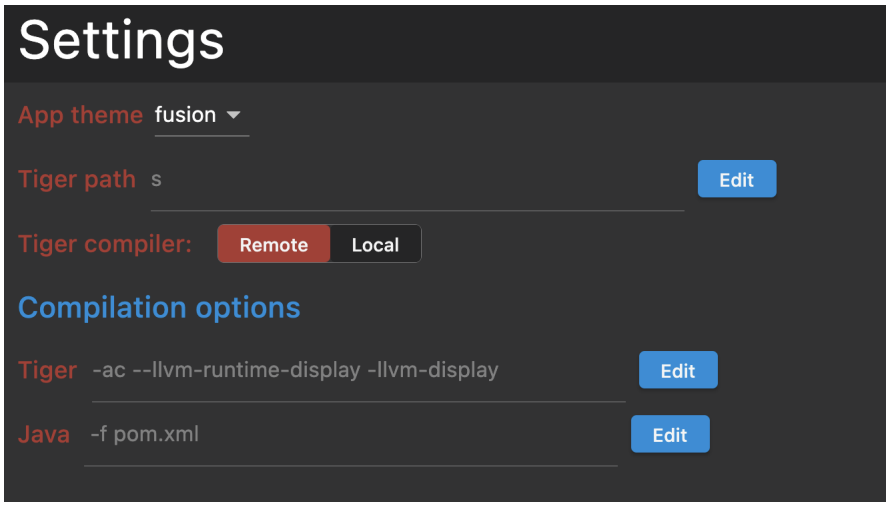

Figure 7: A first look at the settings page

7.1 App theme

You can select your theme between Tiger, Java or Fusion. It will modify the colors of the interface and syntax highlighting, as well as the editor's background.

7.2 Selecting Tiger compiler

You can chose whether you want the IDE to use a remote Tiger compiler or one that you provide locally. If you select the latter, do not forget to provide the path to your compiler in the `Tiger path` section.

Also, if you plan to use the remote compiler, please note that you need a stable internet connection.

7.3 Compilation options

You can precise the compilation options for both Tiger and Java. Please note that the Java code is executed with `mvn exec:java` and then the options.

8 Troubleshooting and FAQ

8.1 Troubleshooting

In this section, we will address some common issues and provide troubleshooting tips for resolving them.

8.1.1 Issue: Can't open file after closing it

- Open an other file.
- You can open the file you closed again.
- We will fix this issue in the next release.

8.1.2 Issue: Can't change the location of new project

- Go back to the `Starter Page` or the `Workspace`
- Click on `New Project` and follow the instruction to create a project.
- We will fix this issue in the next release.

8.1.3 Issue: Code execution errors or unexpected behavior

- Review your code logic and algorithm for potential errors or logical mistakes.
- Check your input data or test cases to ensure they are valid and cover edge cases.
- Consult language-specific documentation or online resources for troubleshooting common runtime errors or unexpected behaviors.

8.2 Frequently Asked Questions (FAQ)

In this section, we will address some frequently asked questions about Jaguar.

8.2.1 Q: Can I customize the Jaguar interface and color scheme?

Yes, Jaguar provides customization options for the interface, that you can select in the settings. You can personalize the IDE's appearance to suit your preferences.

8.2.2 Q: Does Jaguar support version control systems like Git?

Not yet. However we can add in the following releases a Git integration in Jaguar if you feel that it is valuable.

8.2.3 Q: How can I report a bug or suggest a feature for Jaguar?

If you encounter a bug or have a feature request, you can send us an email and we will reach you back. You can find our emails at the end of the user guide.

9 Appendix

9.1 Glossary of Terms

This appendix provides a glossary of key terms and terminology used in Jaguar to help you better understand the software and its features.

- **Integrated Development Environment (IDE):** A software application that provides comprehensive tools and features for software development, including code editing, debugging, and project management.
- **Java:** A widely used programming language that is also supported by Jaguar.
- **Jaguar:** The name of the integrated development environment discussed in this user guide.
- **Tiger:** A programming language specifically designed for use with Jaguar, known for its simplicity and readability.

9.2 Keyboard Shortcuts

Keyboard shortcuts in Jaguar can greatly enhance your productivity by allowing quick access to various features and commands. Refer to the Jaguar documentation for a comprehensive list of available keyboard shortcuts.

Here are some commonly used keyboard shortcuts:

- **Ctrl + S (Windows/Linux) or Cmd + S (macOS):** Save the current file.
- **Ctrl + C (Windows/Linux) or Cmd + C (macOS):** Copy the selected text.
- **Ctrl + V (Windows/Linux) or Cmd + V (macOS):** Paste the copied text.
- **Ctrl + Z (Windows/Linux) or Cmd + Z (macOS):** Undo the last action.
- **Ctrl + A (Windows/Linux) or Cmd + A (macOS):** Select all the text.

9.3 Contact Information

If you have any further questions, need technical support, or wish to provide feedback, please contact our development team using the following information:

- **Alexandra Delin:** alexandra.delin@epita.fr
- **Arnaud Lecoq:** arnaud.lecoq@epita.fr
- **Barbora Plašovská:** barbora.plasovska@epita.fr
- **Charles Neyrand:** charles.neyrand@epita.fr
- **Luc Mahoux:** luc.mahoux@epita.fr

We value your feedback and are here to assist you. Don't hesitate to reach out to us with any questions or concerns.