1 The project

The main goal of the project, is to create a program, which can find executables in a given project and run all of them in a deterministic order. The program should recusrively search the project root and will register all executables and scripts, which should be run at the user's command.

The program should support a Graphical User Interface, where the user can select executables or scripts to run, from the ones that the program found by skimming the project. The program should also show any errors, warnings and messages that were raised by the executables the user chose to run. After the first use of the program, it should save the user's choices into a configuration while, from where it'll be able to automatically rerun all the previously chosen scripts.

The first version of the project will be specialised towards standalone python scripts, which should receive all commandline arguments passed to the program.

2 Market competitors

The key focus of the project, compared to other similar tools in the market, is automatic script detection.

2.1 TeamCity – JetBrains

Teamcity software is powerful CI/CD solution for modern DevOps teams made by JetBrains. Created for speeding up the delivery of any software in the most cost-optimal way, with any tech stack, at any scale. Generally a highly configureable CI/CD tool, which also integrates really well with other JetBrains products. The only downside, is that it has to be manually configured which scripts it should execute on deploy and build operations.

2.2 Ansible – Red Hat

Red Hat Ansible Automation Platform is a unified solution for strategic automation. It combines the security, features, integrations, and flexibility needed to scale automation across domains. It however, suffers from the same configuration issues, as does TeamCity.

3 User interviews

Interview 1: Developer 1 (Front-End Developer)

Interviewer: Can you tell me about some information about your job?

Developer 1: I work as a Front-End Developer at a software developer company.

Interviewer: Can you tell me about your experience with tools that execute scripts from repositories?

Developer 1: I often work with various front-end scripts, especially for building and deploying applications. Sometimes, setting up the environment to run scripts can be time-consuming, especially when I have to manually provide arguments or remember past configurations.

Interviewer: How would a tool that automatically detects runnable scripts and allows you to execute them improve your workflow?

Developer 1: It would be really helpful. Automatically detecting scripts and their required arguments would save me the hassle of figuring out what's needed each time. I usually jump between projects, and this would make it easier to focus on coding rather than environment setup.

Interviewer: What features would you like to see in a tool like this?

Developer 1: First, I'd like an intuitive UI. Dragging and dropping folders to see available scripts and their arguments would be cool. Also, saving configurations would be great so that I don't have to input values repeatedly when switching between similar projects.

Interviewer: Do you often run into issues when executing scripts, like not having the right arguments or encountering errors?

Developer 1: Definitely. Error messages can sometimes be cryptic, so it would help if the tool highlighted missing arguments or provided better guidance on what went wrong when a script fails.

Interview 2: Developer 2 (DevOps Engineer)

Interviewer: Can you tell me about some information about your job?

Developer 2: I work as a DevOps Engineer at a software developer company.

Interviewer: How do you currently manage the execution of multiple scripts in your projects?

Developer 2: In DevOps, we deal with many deployment scripts, monitoring scripts, and automation. Usually, I maintain several shell scripts across different repositories. Managing them manually can be a bit chaotic, especially when they need to be executed in sequence or with certain arguments.

Interviewer: Would a tool that automates script detection and execution assist you in your work?

Developer 2: Definitely. Having a tool that can detect all the runnable scripts in a repo and let me choose which ones to run would streamline things a lot. The ability to batch-execute scripts and see detailed logs in one place would be super useful for deployments.

Interviewer: What specific features would you expect in such a tool to make your job easier?

Developer 2: I'd need good logging and error tracking. For example, if a deployment script fails, I want to see the exact step it failed at. Also, running multiple scripts in parallel or in sequence with configurations saved would be ideal.

Interviewer: Do you find it difficult to manage arguments for different scripts?

Developer 2: Yes, sometimes a script has many arguments that aren't well documented, so figuring them out is a pain. If this tool could retrieve argument details automatically and save configurations, that would save me from a lot of troubleshooting.

Interview 3: Developer 3 (Back-End Developer)

Interviewer: Can you tell me about some information about your job?

Developer 3: I work as a Back-End Developer at a software developer company.

Interviewer: How often do you run scripts while working on back-end services?

Developer 3: I run scripts quite frequently—database migrations, cleanup jobs, data processing, and so on. Each has its own set of arguments, and I often find myself running the same script multiple times with slight variations.

Interviewer: Do you think a tool that detects and lists runnable scripts from a folder would benefit your workflow?

Developer 3: That sounds like a time-saver. If the tool can display runnable scripts with their arguments and allow me to execute them with a single click, it would make repetitive tasks much easier.

Interviewer: What problems do you currently face when executing scripts?

Developer 3: One problem is remembering the exact argument syntax, especially for scripts that are not well-documented. Another is tracking errors—some scripts don't output errors in a way that's easy to understand.

Interviewer: How would you want errors and logs to be handled by this tool?

Developer 3: I'd like real-time logs and clear error messages. A log panel that updates as the script runs would be awesome. Also, if there's a failure, it should give a detailed message and suggestions for fixes, like missing dependencies or incorrect argument values.

Interview 4: Developer 4 (Full-Stack Developer)

Interviewer: Can you tell me about some information about your job?

Developer 4: I work as a Full-Stack Developer at a software developer company.

Interviewer: When you're working across both front-end and back-end codebases, how do you manage running various scripts?

Developer 4: It's pretty manual right now. I have scripts for compiling, bundling, database operations, and server tasks. Some are in Node.js, others in Python or shell scripts. Keeping track of all of them is a bit cumbersome.

Interviewer: Would a centralized tool for script execution make your life easier?

Developer 4: Absolutely. A tool that can detect the available scripts, show me which ones are runnable, and execute them would be a massive help. It would also help if it could remember previous argument configurations, so I don't have to re-enter them every time.

Interviewer: How important is the user interface to you when working with such tools?

Developer 4: The UI is important because I prefer simplicity. If it's cluttered or confusing, it adds more stress to an already complicated task. A clean interface with well-organized sections for scripts, arguments, and logs would be ideal.

Interviewer: Do you encounter issues with script errors or debugging? How should the tool handle them?

Developer 4: Yes, errors are pretty common. I'd appreciate real-time feedback with logs that clearly show where something went wrong. It's also important that the tool shows which arguments are optional or required, so I don't waste time troubleshooting.

4 User hypothesis

The following UX Personas may be relevant:

Persona 1: Emma Thompson (Front-End Developer)

Name: Emma Thompson
Age: 29
Job Title: Front-End Developer
Experience Level: 5 years
Technical Skills: HTML, CSS, JavaScript, React, Webpack, Git
Tools Used: VSCode, Chrome DevTools, GitHub, Jenkins
Environment: Working in a fast-paced startup with a lean development team

Goals

- Quickly test, build, and deploy front-end code with minimal setup.
- Have a streamlined development process with a focus on UI/UX improvements.
- Reduce time spent setting up environments or manually configuring scripts.

Frustrations / Pain Points

- Frequently has to reconfigure and manually set up build tools (e.g., Webpack or custom build scripts).
- Time-consuming debugging of environment-related issues when trying to run scripts.
- Lack of automatic detection for arguments or clear error messages when executing tasks.

Needs from the Tool

- A clear, simple UI that allows her to drag and drop folders, detect runnable scripts, and easily input arguments.
- The ability to save previous configurations to avoid repetitive setup.
- Easy access to logs and error messages with actionable feedback.

Motivations

- To focus more on improving the user interface and user experience rather than backend tasks or script management.
- Increase productivity by reducing the need for repeated configuration steps.

Persona 2: David Ruiz (DevOps Engineer)

Name: David Ruiz
Age: 35
Job Title: DevOps Engineer
Experience Level: 10 years
Technical Skills: Python, Bash, AWS, Docker, Jenkins, Kubernetes
Tools Used: AWS CLI, Docker, Jenkins, Terraform, Ansible
Environment: Managing a cloud infrastructure for an enterprise company

Goals

- Automate and streamline the deployment process across multiple environments.
- Ensure high reliability and efficiency in managing automation and script execution.
- Manage multiple repositories containing various scripts for monitoring, deployments, and backups.

Frustrations / Pain Points

- Struggles with manually maintaining and running scripts, especially when managing large-scale deployments.
- Difficult to debug and trace script execution errors across distributed environments.
- Often deals with complex argument configurations for deployment scripts, which are time-consuming to manage.

Needs from the Tool

- An efficient script detection mechanism that automatically lists runnable scripts from different repositories.
- A robust logging and error-handling system to track deployment failures with detailed reports.
- The ability to save and reuse configurations, execute scripts in sequence or parallel, and automate common tasks.

Motivations

- Increase efficiency in managing and executing multiple deployment scripts across cloud environments.
- Ensure system stability and reduce downtime by minimizing manual errors during script execution.

Persona 3: Michael Wong (Back-End Developer)

Name: Michael Wong
Age: 32
Job Title: Back-End Developer
Experience Level: 7 years
Technical Skills: Python, Node.js, SQL, REST APIs, Docker, Git
Tools Used: Postman, MongoDB, MySQL, Jenkins, Docker
Environment: Works in a mid-sized company, maintaining APIs and database services for various clients

Goals

- Optimize database migrations, data processing tasks, and API deployment scripts.
- Automate routine back-end tasks and improve code quality.
- Run recurring scripts with minimal effort and avoid redundant configurations.

Frustrations / Pain Points

- Recurring issues with managing scripts that have complex argument structures.
- Difficulties troubleshooting script errors due to insufficient logging or unclear error messages.
- Struggles with running the same scripts multiple times with different configurations, wasting time re-entering arguments.

Needs from the Tool

- A centralized interface to manage all back-end scripts and execute them with the correct arguments.
- Detailed logs and real-time feedback on script execution to debug failures quickly.
- The ability to load, edit, and save configurations for routine tasks like database migrations or API testing.

Motivations

- To reduce the time spent managing scripts and configurations, freeing up more time to focus on API development.
- Enhance system performance and reliability by automating repetitive tasks with minimal errors.

5 Market position

Following market research, it is evident that no tool currently exists that can locate executables and configure them in a manner that we want it to. That's the main reason, why our product can be more usable for developers. Developers can run their projects and store their configurations in one place, on a user friendly interface. There is no need to use command line prompt or create configuration files manually.

Our project could take a load off of the shoulders of people, who work in sectors where automating labourous tasks is important. This could help companies increase their productivity in multiple departments.

6 Links

[1] - TeamCity - JetBrains
[2] - Ansible - Red Hat