INNO 2 - Video Script

Slide 1

Hello and welcome to a short presentation of our Innovation Lab project. First, let me introduce our team. We are a team of three consisting of Fatema Aly, Jan Loos and yours truly Taha Samaha. We have dealt with the topic of PHP linters and now we are going to go into more detail about the goals we had and have achieved.

Slide 2

We were given the task of searching for two compatible PHP linter and then compiling them into one. The programming language to be used is Python. Thanks to the research from last semester, we already have a few linters to choose from and know their strengths and weaknesses. The aim of the two linters is to check PHP files in one go and output the result in an HTML/PHP document. If there are duplicates in the output, they should be reduced to one entry. We didn't forget the Moodle design pattern and adapted the CSS properties to it.

Slide 3

We chose PHPMD and PHPSTAN because of the compatibility, efficiency and diversity of the rulesets.
PHPMD is a spin-off project of PHP Depend and aims to be a PHP equivalent of the well known Java tool PMD. It can be seen as an user-friendly frontend application for the raw metrics stream measured by PHP Depend.
PHPStan focuses on finding errors in your code without actually running it. It catches whole classes of bugs even before you write tests for the code. It moves PHP closer to compiled languages in the sense that the correctness of each line of the code can be checked before you run the actual line.

Slide 4

Let's watch a live demo and see how it works!
[Input Folder]
Let's take a look at our work environment, the first thing that's very important is our input file, which we're going to lint. It consists of ten functions where errors have been introduced.
We call the actual linters with just one phar file without any presettings. In this case we use the PHPMD and the PHPSTAN Phar file.
[run.py]
In the python file, we merge both Lint processes and exclude duplicates. With the package dominate we manipulate HTML elements and their attributes. For example here, I create a heading 1 with the text-content “result” and the class “container”.
[Output Folder]
We save the result in the output folder as out.html, where we have already prepared a CSS file that adapts the output to the Moodle design pattern.
[Browser - out.html] **PAUSE - CHANGE VIEW**
The end result then looks like this and we see the following elements:
The Path of the Linted File

- Here on the left side we see:
    - Which Linter throwed the entry.
    - the triggered ruleset
    - the line number
- And on the right we see the content of the error message

Slide 5

Now for our retrospective…
Originally, the goal of the project was to develop an improvement or an extension for the Moodle plugin EspressoIT to support additional code quality and safety functions for PHP files.
Unfortunately, EspressoIT was turned off and our task has changed.
However, we made the best of it and were always on time with the sprints and we have successfully completed the task given to us. Of course, we also adhered to the FH scheme.
Along the way, we learned a lot about linters and how they exactly work.

Slide 6

Here are some useful links to our project and sources we used.