



Python Session 6

## This session

1. Overview of projects
2. Form teams
3. Introduction to GitHub and sharing code
4. Guide to API keys
5. Planning your project
6. Project work

## **Project Briefs**

Until the end of the course you will be working on team projects

In the second half of the final session each team will present their project

In your student handbook there are the different project briefs

Each brief includes:

- Overview of the project
- Required tasks or ideas for data analysis
- Example project code

When you start the project, do the required tasks first. After completing the required tasks, add your own ideas to the project

Two types of project are available:

- API Based projects
- Data analysis projects

API Based projects:

- Top Trumps Game
- Cryptocurrency Converter
- Recipe Generator



## Data analysis projects:

- IMDb movie data
- Crime data: London 2019 or San Francisco 2018
- University Rankings data
- World Food Production 2018 data
- World Happiness data 2015 & 2016

**Teams**

**Exercise 6.1:** Get into teams of 2-3 people. The instructors will create a Slack channel for each group and add you to it.

# **Introduction to GitHub and sharing code**

# API key setup in PyCharm

## *What is an API key?*

API keys and IDs are used to monitor and control how the API is being used by a specific user.

If you are doing a Project which involves an API, you may need to create an account and get an API key, and sometimes an API ID too. You will need this to access the API data.

## *Keeping your Keys private*

You should treat API keys and IDs like your other passwords- they should always be kept private.

Some APIs charge for their use- you may pay monthly for a certain number of requests, or pay per request. If people have your keys, they can make requests to the server using your credentials.

In order to keep the keys private, we will never directly write them in our Python code, but will store them as 'Environment variables'.

Each person in the group should generate their own API keys. Never send your API keys to GitHub, as they may be publicly accessible.

## Environment Variables

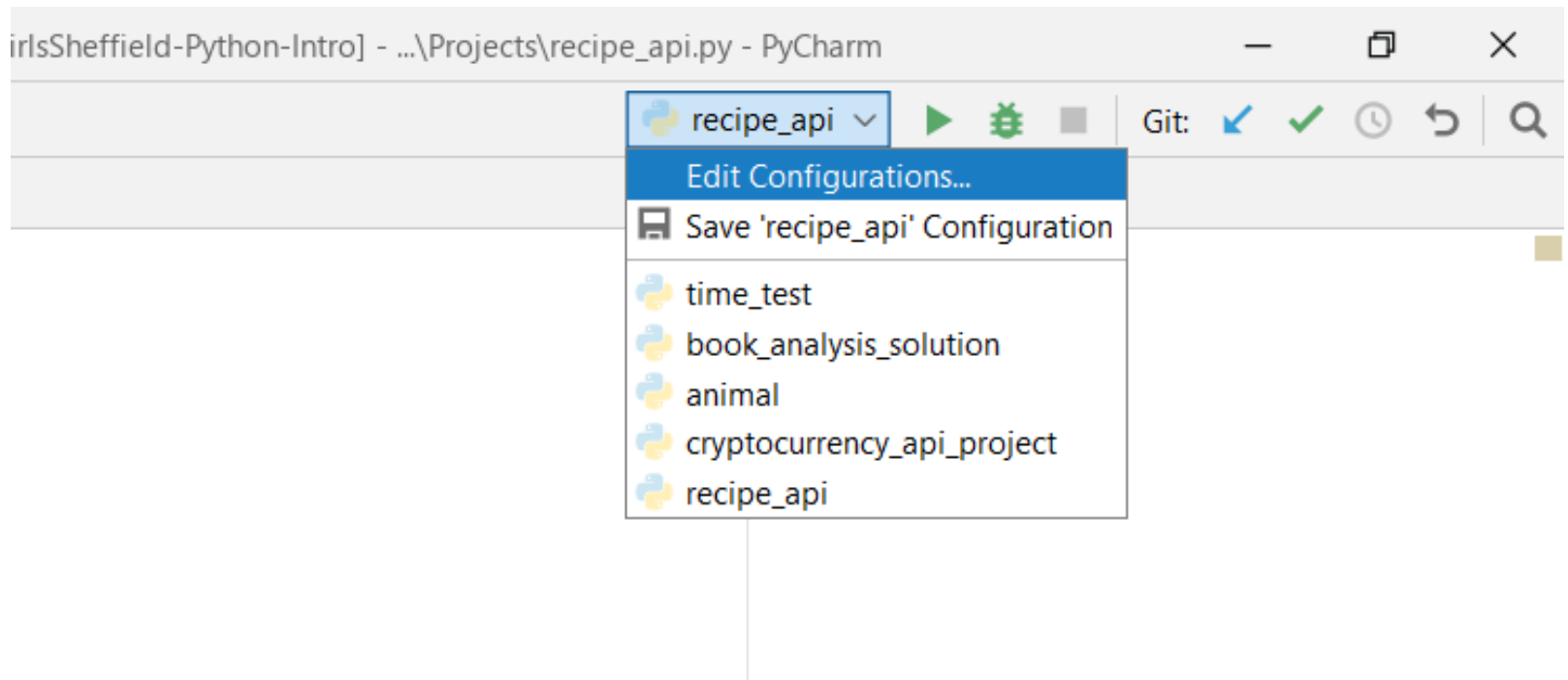
Environment variables are variables that exist outside of our code files. We can access them to use in our programs.

They exist in 'key' and 'value' pairs. Each value has an associated key which can be called from our python code to access the value data.

For example our key name might be `recipe_api_key` and the associated value will be our API key: eg. `34hddm44ksaFtG`

## Adding API keys to PyCharm

*Step 1: Navigate to and open 'Edit Eonfigurations' for your file*





## ***Step 2: Open environment variables***

Name:  ☐ Share through VCS ☐ Allow parallel run

Configuration Logs

Script path:

Parameters:

Environment

Environment variables:

Python interpreter:

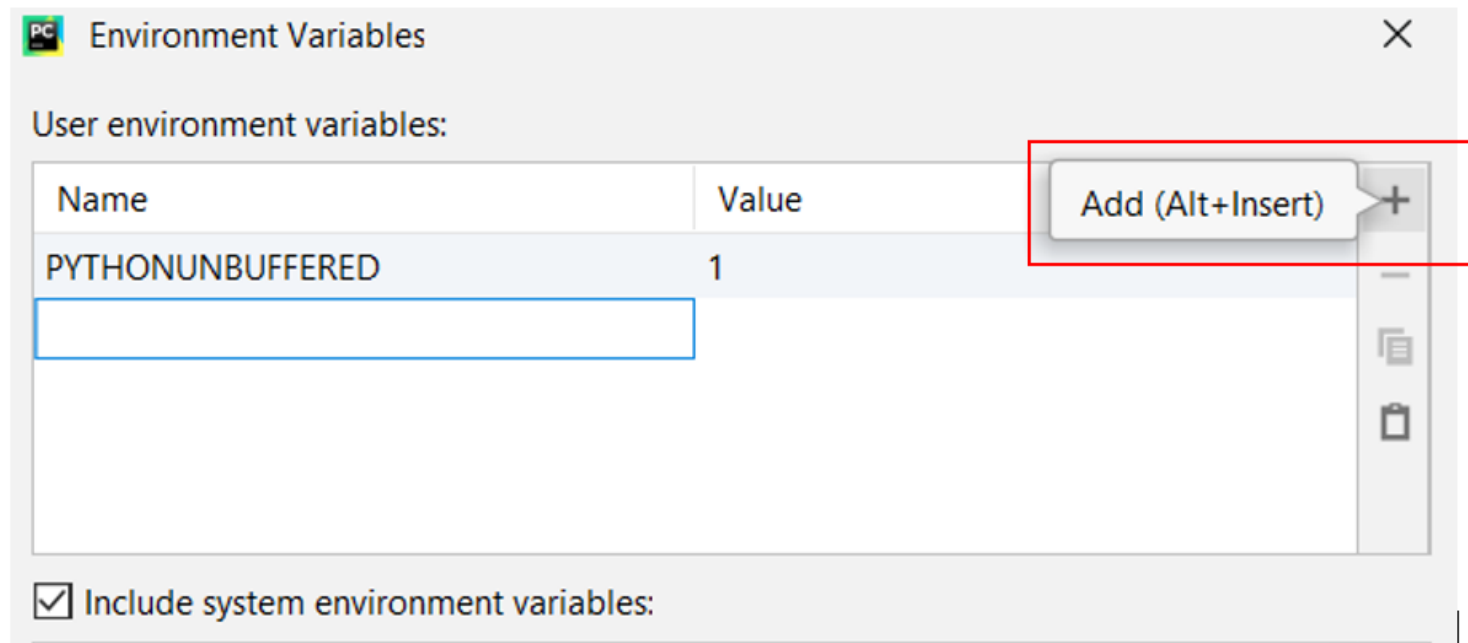
Interpreter options:

Working directory:

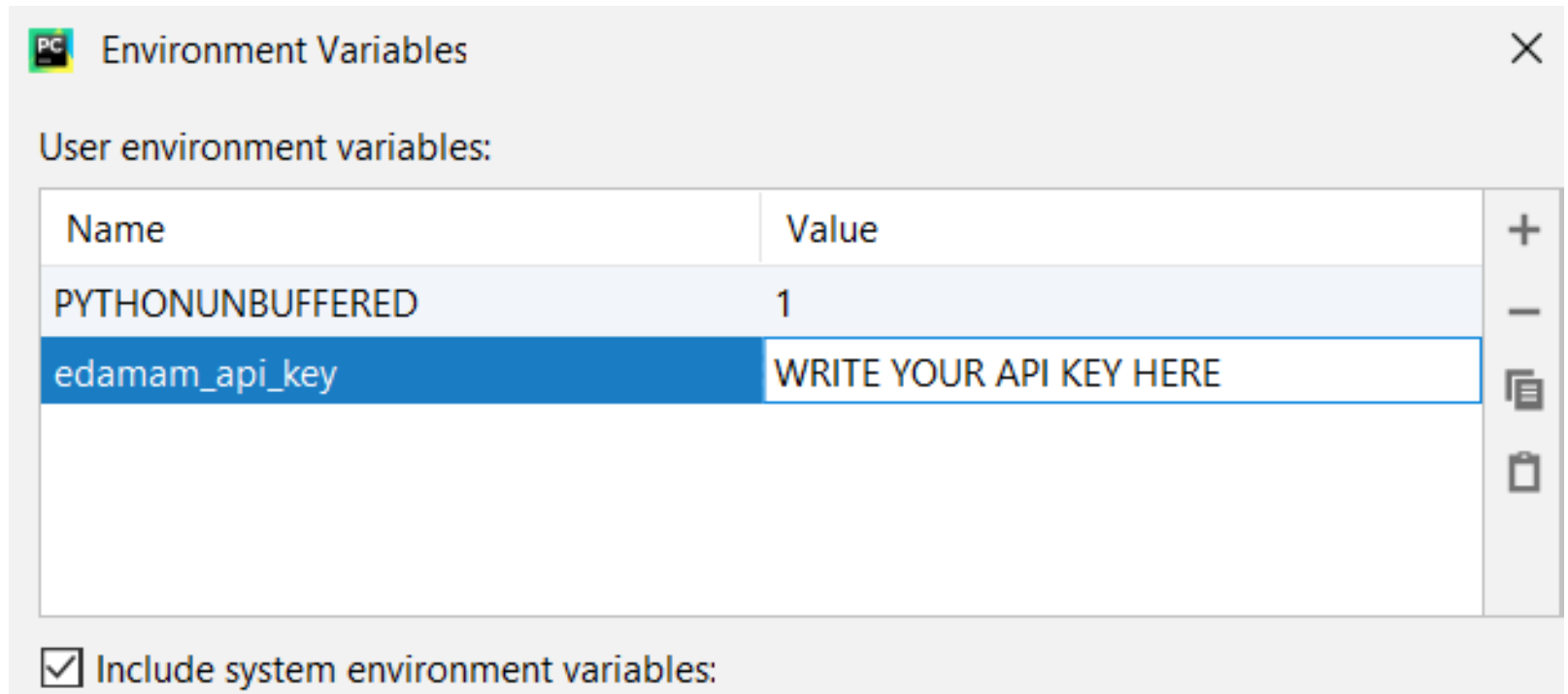
☒ Add content roots to PYTHONPATH

☒ Add source roots to PYTHONPATH

***Step 3: Select 'Add' to insert a new Environment Variable***



***Step 4: Add your API Keys and API ID (if required)***



The screenshot shows the 'Environment Variables' dialog box in Windows. The title bar includes a 'PC' icon and a close button. The main section is titled 'User environment variables:'. Below this is a table with two columns: 'Name' and 'Value'. The table contains three rows: a header row, a row for 'PYTHONUNBUFFERED' with value '1', and a row for 'edamam\_api\_key' with value 'WRITE YOUR API KEY HERE'. The 'edamam\_api\_key' row is highlighted in blue. To the right of the table are icons for adding (+), removing (-), and copying (document icon) variables. At the bottom, there is a checkbox labeled 'Include system environment variables:' which is checked.

Name	Value
PYTHONUNBUFFERED	1
edamam_api_key	WRITE YOUR API KEY HERE

☒ Include system environment variables:

### *Step 5: Restart PyCharm and import the API keys*

We can import environment variables as follows:

```
import os  
  
api_id = os.environ.get("edamam_api_key")  
api_key = os.environ.get("edamam_api_key")
```

We now have our API keys and ID stored as variables.

# Planning Requirements

After completing the project's required tasks, you can customise it with your own ideas

MoSCoW is a technique for prioritising requirements

Requirements are put into four categories:

- Must
- Should
- Could
- Won't

A example list of requirements for a book search:

- Users should be able to enter the book's name to search for it
- The price of the book should be shown on the search results
- The user should be able to order the book from Amazon
- The author of the book should be shown in the search results
- The book's front cover should be shown in the search results



Prioritised list of requirements:

Must

- Users should be able to enter the book's name to search for it
- The author of the book should be shown in the search results

Should

- The price of the book should be shown on the search results

Could

- The book's front cover should be shown in the search results

Won't

- The user should be able to order the book from Amazon

**Exercise 6.2:** Come up with ideas for your project.

1. Individually, come up with as many ideas for the project as possible
2. Share one of your ideas with the team, going around each team member until all ideas have been covered
3. As a team, prioritise the ideas into "Must, Should, Could, Won't"

**Project Work**

Work on your project with your team

Tips:

- Write your code as functions
- Share your code with other team members frequently
- Try to break down big tasks into smaller chunks

## Session Recap

**Question 1:** Name the four categories for MoSCoW

**Homework:** Continue working on your projects

Next week: Time to work on your projects with help from instructors