

**Installing Python**

**Handbook Guide**

February 2023

Contents

Python Installation

* Installing Python using Anaconda
* Verify Python Installation

Managing environtments

* Create Environtment
* Activate Environtment
* Installing Library
* Export env.yml
* Remove environtment
* Create env.yml

Installing Notebook Extension

* Install and Manage NB Extensions

Link refrence : <https://conda.io/projects/conda/en/latest/user-guide/tasks/manage-environments.html#activating-an-environment>

**Python Installation**

**Menginstal Python menggunakan Anaconda**

Untuk instalasi python, disini kami akan menggunakan serta menginstal paket manajer bernama ***Anaconda***. Dengan Anaconda, pengguna tidak hanya mendapatkan Python yang terinstal tetapi juga akan memiliki package yang diperlukan (i.e. numpy, pandas)., Anaconda juga akan menyertakan penginstallan IDE(Integratet Development Environtment) seperti ***Spyder*** dan ***VSCode***,jadi setelah membuka tautan di bawah ini, silakan pilih Python versi 3.9 untuk instalasi **Python version 3.9 for installation**.

Use this link: <https://www.anaconda.com/products/distribution>

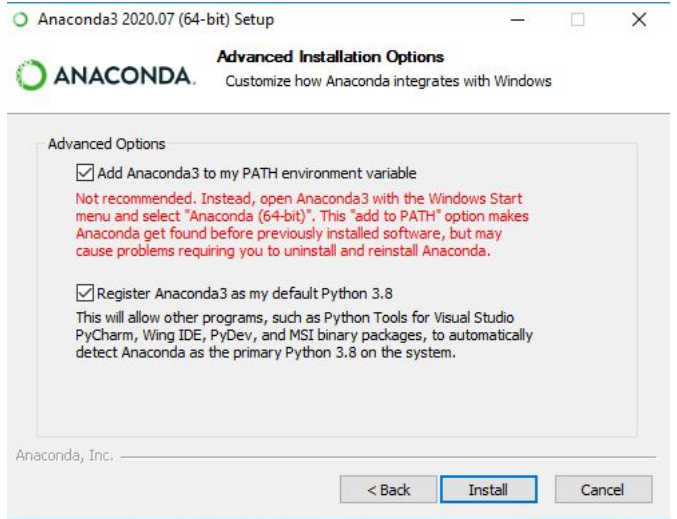
* Pilih Sistem Operasi yang sesuai, pastikan Anda memilih versinya yang kompatibel dengan kecepatan bit komputer anda.



Installing Anaconda



Gambar 1.0 Installing Anaconda



Gambar 1.1 Installing Anaconda

Untuk pengguna **windows** pastikan Anda mencentang **Add Anaconda to my PATH to my PATH environtment variable** dan kemudian tunggu sampai instalasi selesai.

**Managing Environtment**

Environment pada Anaconda Prompt adalah sebuah lingkungan virtual yang dapat dikonfigurasi untuk memungkinkan kita menginstal dan menggunakan package Python dan program lainnya tanpa mengganggu konfigurasi sistem operasi kita secara keseluruhan.

Setiap environment memiliki versi Python yang terisolasi, dan package yang dapat diinstal tergantung pada versi Python tersebut. Dengan cara ini, kita dapat menginstal dan mengelola package Python dengan cara yang terpisah untuk setiap proyek kita. Dengan menggunakan environment, kita dapat memastikan bahwa setiap proyek memiliki lingkungan yang terpisah, sehingga mencegah terjadinya konflik package yang berbeda.

Misalnya, jika kita ingin membuat proyek menggunakan package A, tetapi package A memerlukan versi Python yang berbeda dengan yang terinstal pada sistem operasi kita, maka kita dapat membuat environment baru dengan versi Python yang dibutuhkan oleh package A.

Dalam keseluruhan, environment pada Anaconda Prompt adalah alat yang berguna untuk mengelola package dan lingkungan Python pada proyek yang berbeda, dengan cara yang terpisah dan terisolasi. Dengan menggunakan environment, kita dapat menghindari masalah konflik package dan memastikan konsistensi lingkungan pada setiap proyek kita.

**Lalu apa itu Python dan Anaconda?**

Python dan Anaconda keduanya berkaitan dengan bahasa pemrograman Python, namun keduanya memiliki perbedaan yang mendasar:

1. Python: Python adalah bahasa pemrograman tingkat tinggi yang sering digunakan dalam pengembangan aplikasi dan analisis data. Python dapat diunduh dari situs web resmi python.org dan dapat diinstal secara terpisah pada sistem operasi yang berbeda.
2. Anaconda: Anaconda adalah distribusi Python yang populer, yang mencakup interpreter Python, package, dan environment manajemen yang dapat membantu pengguna untuk memasang dan mengelola package Python. Anaconda hadir dengan lebih dari 1000 package data science populer, termasuk NumPy, Pandas, Matplotlib, Scikit-learn, dan sebagainya. Selain itu, Anaconda juga menyediakan lingkungan virtual yang memungkinkan pengguna untuk membuat dan mengelola lingkungan Python terisolasi secara independen, sehingga dapat menghindari masalah konflik package yang mungkin terjadi.

Dalam istilah sederhana, kita dapat menganggap Python sebagai bahasa pemrograman dasar, sedangkan Anaconda adalah platform yang terdiri dari bahasa Python dan package-package yang dapat digunakan untuk pengembangan aplikasi dan analisis data.

Dengan demikian, perbedaan utama antara Python dan Anaconda adalah bahwa Python hanya menyediakan interpreter dan beberapa package dasar, sedangkan Anaconda menyediakan banyak package data science populer dan lingkungan manajemen package yang lebih mudah. Anaconda sangat populer dalam pengembangan aplikasi dan analisis data, terutama bagi mereka yang memulai dengan Python dan perlu menginstal banyak package dan environment untuk mengatasi kebutuhan mereka.

**Lalu apa itu package dan library?**

Secara umum, **package** dan **library** pada Python dapat dianggap sebagai hal yang sama, yaitu kumpulan modul atau kode yang berisi sekumpulan fungsi dan fitur untuk menyelesaikan tugas tertentu dalam bahasa pemrograman Python.

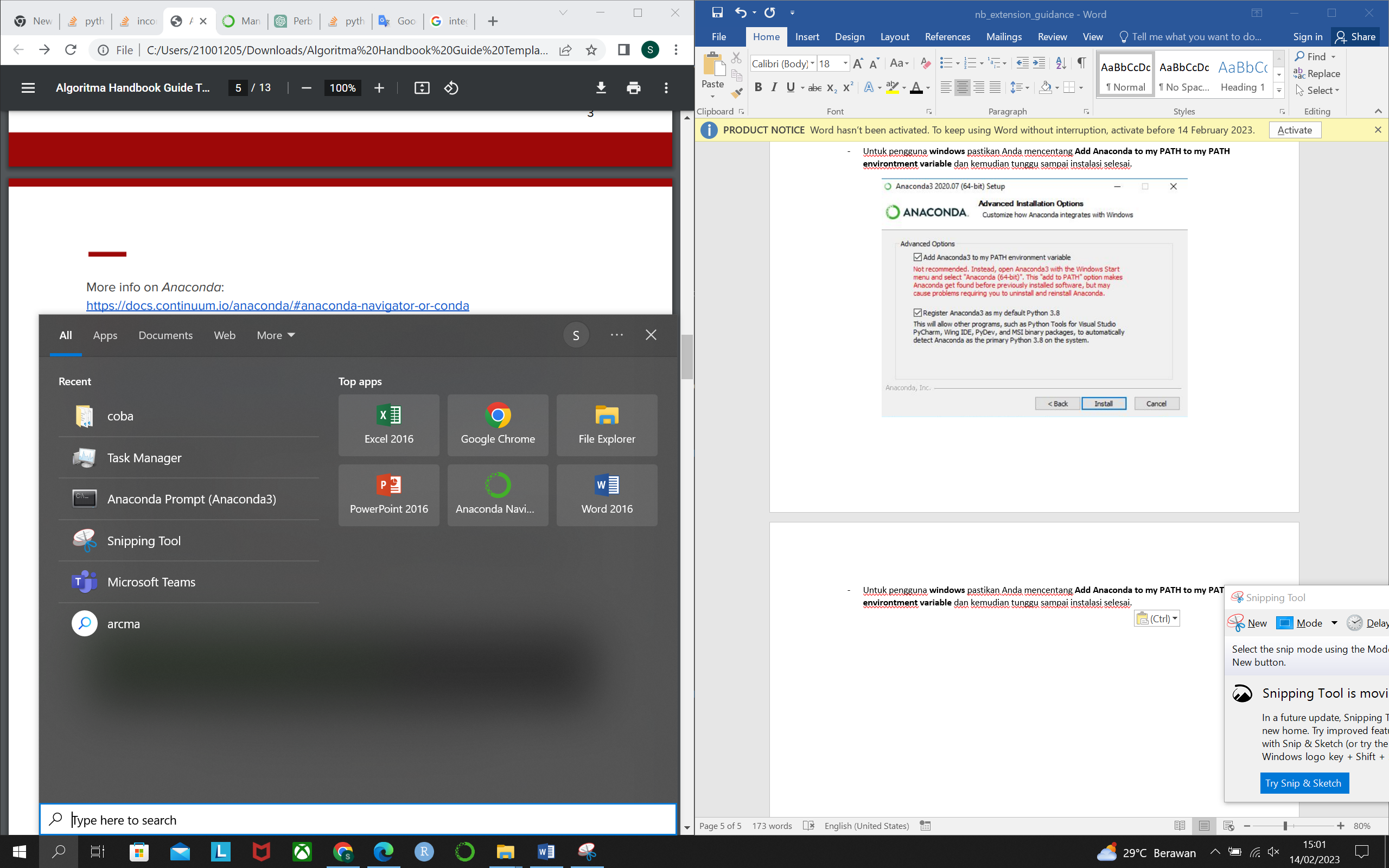
Namun, ada perbedaan kecil antara keduanya. **Library** adalah kumpulan modul yang menyediakan fungsionalitas tertentu, seperti NumPy, Pandas, Matplotlib, dan sebagainya. Library biasanya sudah tersedia di instalasi Python, dan dapat diimpor ke dalam proyek dengan menggunakan perintah **import**.

Sementara itu, **package** adalah kumpulan dari satu atau lebih library dan modul terkait lainnya, yang biasanya digunakan untuk menyelesaikan tugas yang lebih spesifik. Package seringkali berisi beberapa library dan modul yang saling terkait, dan dapat diinstal secara terpisah dari instalasi Python utama. Contoh package populer di Python termasuk Django, Flask, dan Tensorflow.

Dalam istilah sederhana, kita dapat menganggap library sebagai buku dalam perpustakaan dan package sebagai rak yang berisi beberapa buku atau topik terkait.

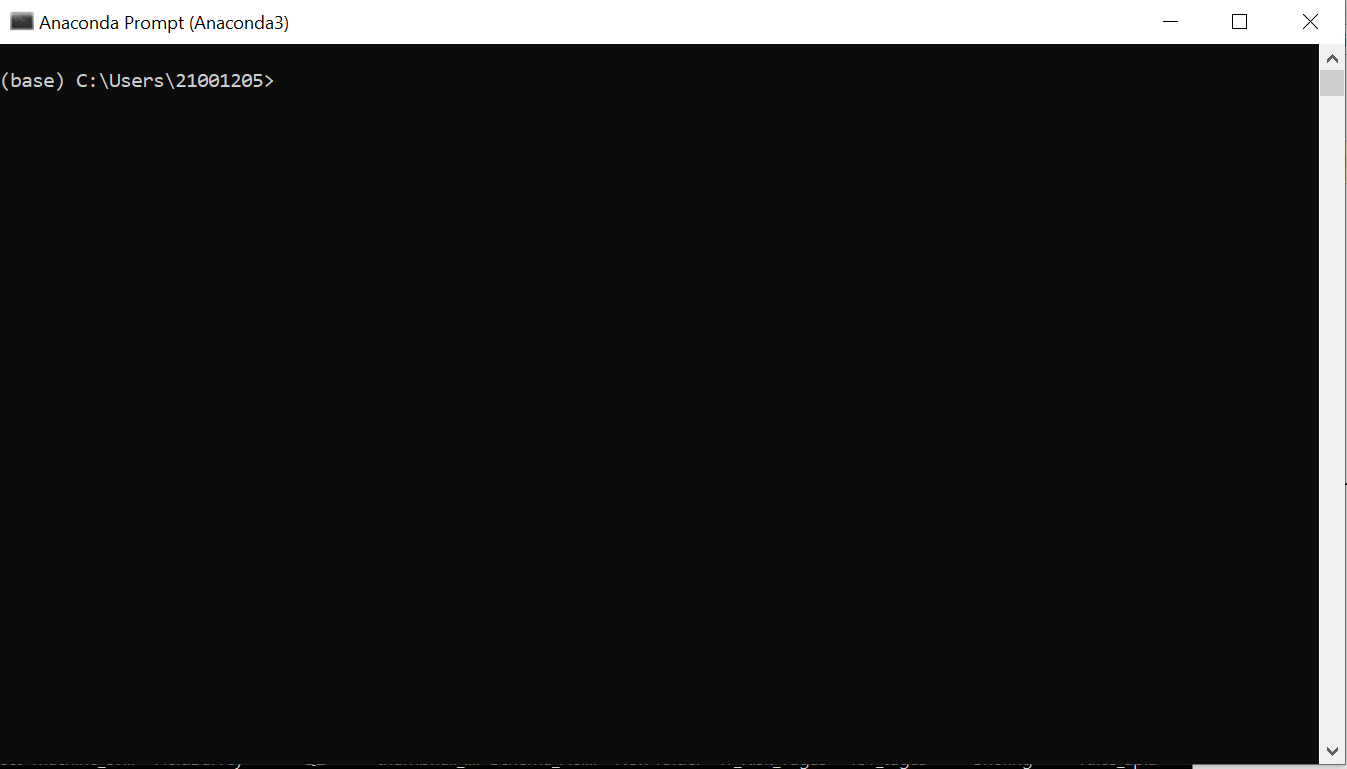
Dalam Python, penggunaan package dan library sangat umum dalam pengembangan aplikasi dan analisis data. Kedua hal tersebut sangat penting karena memungkinkan pengguna untuk menggunakan modul dan fungsi yang sudah ada, sehingga dapat menghemat waktu dan upaya dalam membuat kode dari awal.

Lanjut dari step yang sebelumnya, setelah Anaconda terinstall anda bisa menjalankan python pada **anaconda navigator** langsung atau menggunakan **anaconda prompt,** namun pada course ini saya akan mencoba membahas untuk menjalankannya pada **anaconda promt** terlebih dahulu , mengingat kedua step selanjutnya akan lebih efisien jika dilakukan pada **anaconda prompt**.



Gambar 2.0 Chose Anaconda Prompt(Anaconda3)

Klik menu windows di ujung layar kemudian pilih Anaconda Prompt, lalu tahap selanjutnya akan mencoba create environtments baru serta menginstall dan memanage notebook exstension



Gambar 2.1 Anaconda Prompt(Anaconda3) **Base Environtment**

Pada tahapan awal anda akan berada pada base environtment, adapun kagunaan dan alasan membuat environment baru pada Anaconda Prompt yaitu:

Ketika anda bekerja pada beberapa proyek atau memerlukan beberapa versi Python dan dependensi yang berbeda secara bersamaan. Beberapa alasan mengapa anda harus membuat environment baru pada Anaconda Prompt antara lain:

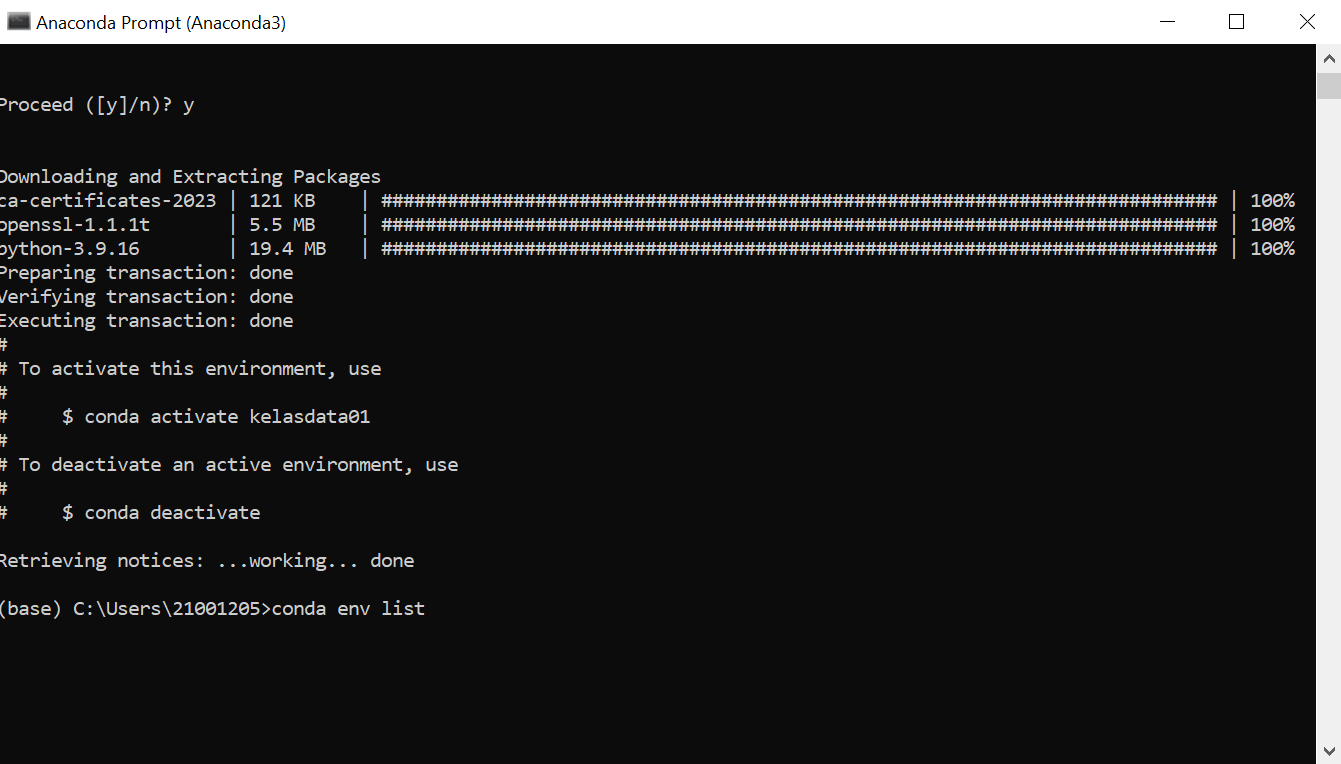
* Membuat lingkungan yang bersih: Dengan membuat environment baru, Anda dapat memulai dari lingkungan yang bersih dan terisolasi, di mana Anda dapat menginstal versi Python, package, dan dependensi yang diperlukan untuk proyek tertentu tanpa mempengaruhi environment atau dependensi yang sudah ada.
* Menghindari konflik: Dalam beberapa kasus, package atau dependensi yang diinstal untuk satu proyek dapat menyebabkan konflik dengan package atau dependensi yang diinstal untuk proyek lain. Dengan membuat environment baru untuk setiap proyek, Anda dapat menghindari konflik ini dan memastikan bahwa setiap proyek memiliki environment yang terisolasi dan stabil.
* Mudah dikelola: Dengan membuat environment baru pada Anaconda Prompt, Anda dapat mengelola environment dan dependensi proyek Anda secara terpisah, membuatnya lebih mudah untuk menambahkan, menghapus, atau memperbarui package dan dependensi tanpa mempengaruhi lingkungan lain.
* Reproducible: Dengan membuat environment baru, Anda dapat membuat lingkungan yang dapat direproduksi di lingkungan lain, memastikan bahwa proyek Anda dapat berjalan pada lingkungan yang sama bahkan jika lingkungan tersebut terpisah secara fisik.



Gambar 2.2 Anaconda Prompt(Anaconda3) **Create new enirontment with Python version 3.9**

Gambar diatas merupakan tahap pembuatan environtment baru dengan **nama environtment kelasdata01** dan dengan **versi python 3.9 ( conda create –n kelasdata01 python=3.9 )**

Setelah berhasil membuat environtment baru maka tahap selanjutnya adalah melihat ada berapa environtment yang terdapat pada computer ini, dengan cara mengetikan ( **conda env list** )



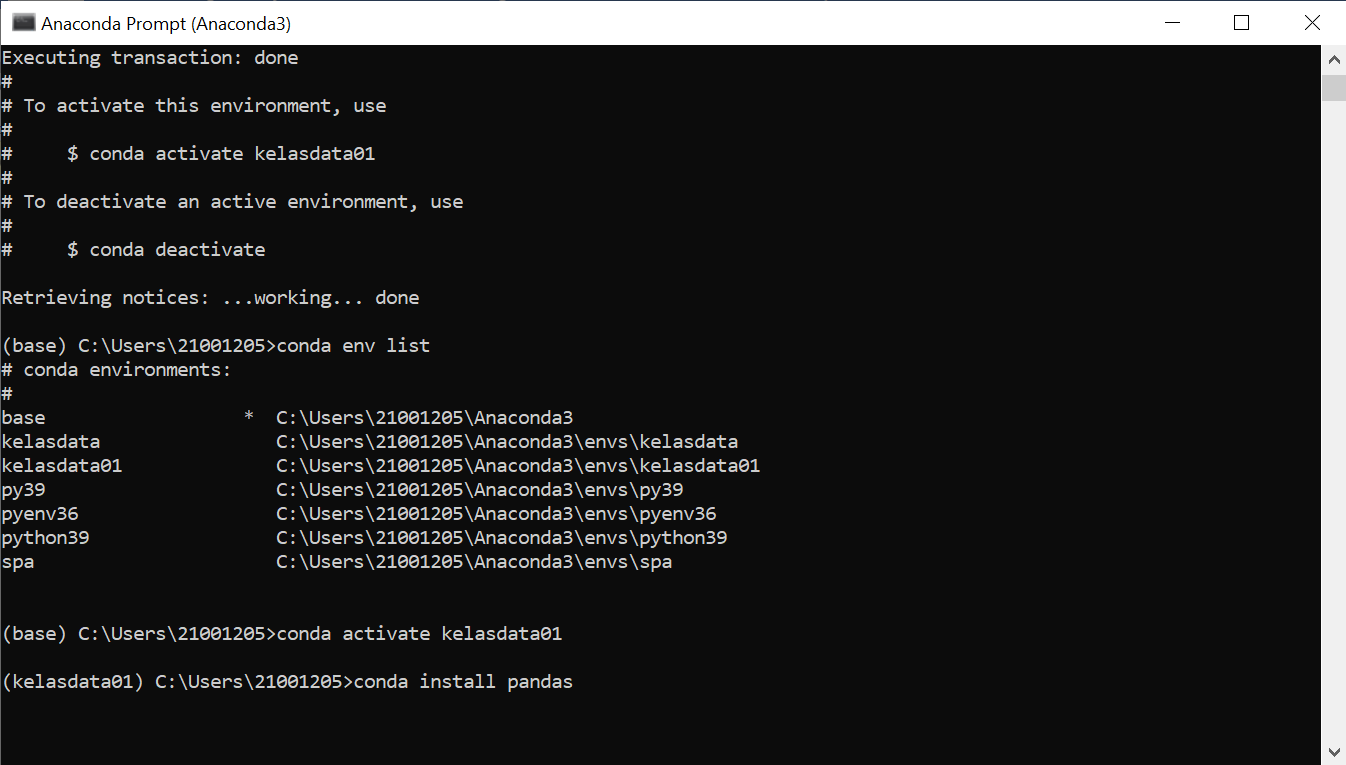
Gambar 2.3 Anaconda Prompt(Anaconda3) **melihat daftar environtment yang tersedia**

Gambar dibawah pada garis merah merupakan daftar environtment apa saja yang ada pada computer, dan pada garis biru merupakan tahapan aktivasi environtment yang telah dibuat sebelumnya dengan cara mengetikan ( **conda activate kelasdata01 ) ,** dapat terlihat perpindahan environtmentnya dari **(base) ke (kelasdata01).**



Gambar 2.4 Anaconda Prompt(Anaconda3) **Aktivasi environtment from base to kelasdata01**

Setelah berhasil mengaktivasi environtment, pada tahapan selanjutnya akan coba dilakukan penginstallan library pandas dengan mengetikan ( **conda install pandas** )



Gambar 2.5 Anaconda Prompt(Anaconda3) **Installing Library**

**Lalu apa itu library dan mengapa kita harus menginstall nya?**

**Library** pada Python adalah kumpulan modul atau kode yang berisi sekumpulan fungsi dan fitur untuk menyelesaikan tugas tertentu dalam bahasa pemrograman Python. Library seringkali berisi fungsi-fungsi yang terkait dengan topik tertentu, seperti Matplotlib untuk visualisasi data, Pandas untuk analisis data, atau Scikit-learn untuk machine learning.

Mengapa kita harus menginstal library? Ketika kita bekerja dengan Python, seringkali kita membutuhkan fungsi-fungsi yang tidak tersedia secara default dalam instalasi Python dan tidak memungkinkan semua itu diprogram manual secara eksplisit. Itulah mengapa kita perlu menginstal library agar dapat menggunakan fungsionalitas tambahan yang tidak tersedia secara default.

Contoh sederhana, saat kita ingin memvisualisasikan data dalam bentuk grafik, kita dapat menggunakan library Matplotlib. Tanpa Matplotlib, kita harus membuat grafik dari awal dengan menggunakan kode Python, yang sangat memakan waktu dan tidak efisien.

Ada ribuan library Python yang tersedia, masing-masing dengan fungsionalitas dan tujuan yang berbeda. Beberapa library populer diantaranya NumPy, Pandas, Scikit-learn, TensorFlow, Keras, dan Pygame. Untuk menginstal library, kita dapat menggunakan package manager seperti pip atau conda.

Jadi, kesimpulannya, kita perlu menginstal library agar dapat menggunakan fungsionalitas tambahan dalam Python dan menghemat waktu dan usaha dalam menulis kode dari awal.

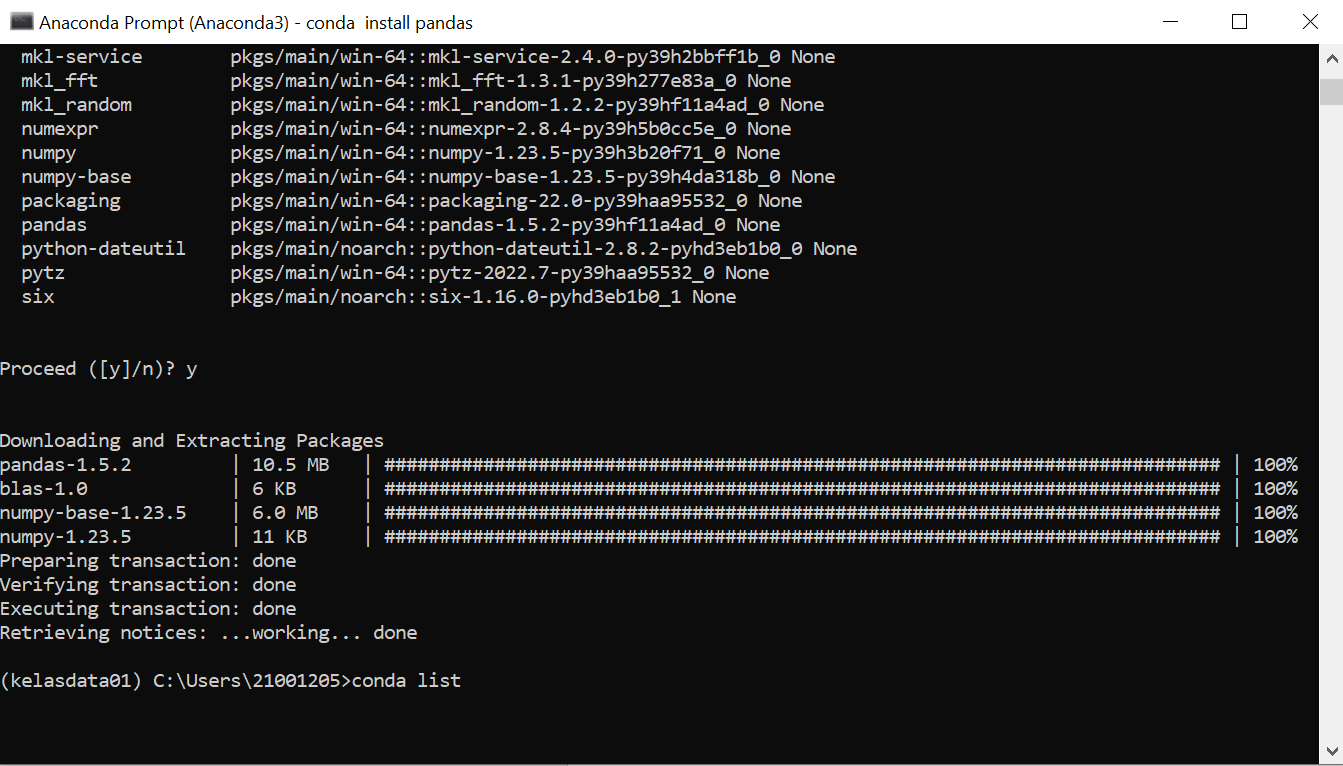
**Apa bedanya Conda Install dengan Pip Install?**

**Pip** adalah manajer paket Python default, yang mengelola paket Python yang diinstal di sistem dan dependensinya. Pip hanya dapat mengelola dependensi Python dan tidak dapat mengelola dependensi non-Python secara otomatis. Jika memerlukan paket Python dan dependensi non-Python, pengguna harus menginstal dependensi non-Python secara terpisah. Pip juga tidak menyediakan alat untuk mengelola environment, dan pengguna harus menggunakan alat lain, seperti virtualenv atau venv, untuk membuat dan mengelola environment Python, Pip juga biasanya digunakan untuk menginstal package yang tidak tersedia dalam distribusi Anaconda atau tidak terkait dengan pengembangan ilmiah atau analisis data.

Sementara itu, **Conda** adalah manajer paket yang lebih kompleks dan canggih, yang dapat mengelola paket Python dan non-Python, serta dependensi mereka. Conda juga menyediakan alat untuk mengelola environment, yang memungkinkan pengguna untuk membuat dan mengelola environment yang berbeda dengan versi Python dan dependensi yang berbeda-beda. Selain itu, Conda juga dapat mengintegrasikan beberapa pustaka atau aplikasi tertentu yang diinstal, dengan fitur seperti jupyter notebook, spyder dan sebagainya.

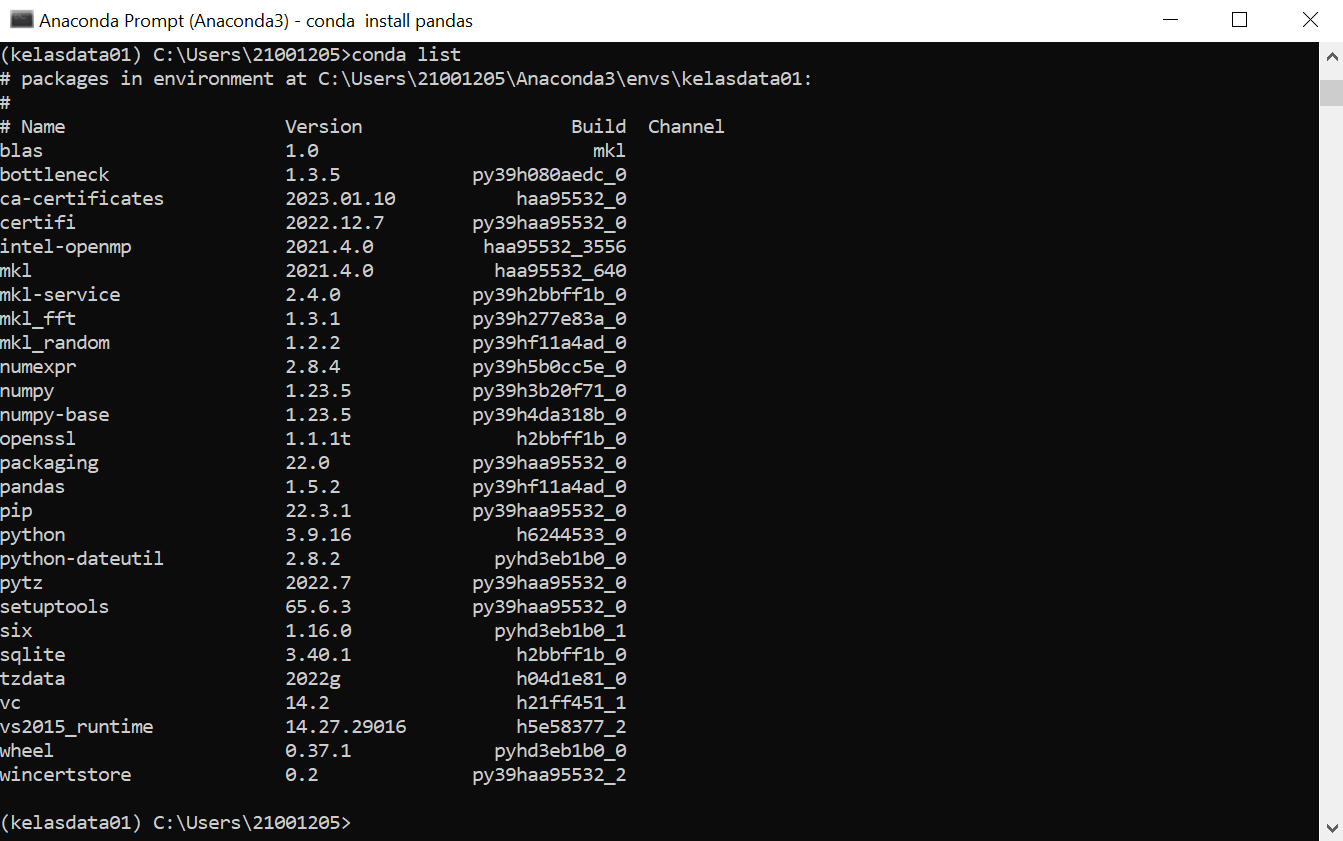
Dengan demikian, perbedaan paling signifikan antara pip dan conda adalah pada kemampuan Conda untuk mengelola dependensi non-Python secara otomatis dan menyediakan fitur untuk mengelola environment. Namun, terkadang Conda tidak memiliki paket terbaru yang tersedia pada PyPI, dan dalam beberapa kasus, pip mungkin masih menjadi pilihan yang lebih tepat. Pilihan manajer paket tergantung pada kebutuhan proyek dan preferensi pengguna.

Kesimpulannya, Conda lebih disarankan untuk proyek-proyek pengembangan ilmiah dan analisis data, sedangkan Pip lebih disarankan untuk proyek-proyek umum yang berkaitan dengan Python. Namun, pada akhirnya, pilihan alat manajemen package yang tepat tergantung pada kebutuhan dan tujuan proyek masing-masing.



Gambar 2.6 Anaconda Prompt(Anaconda3) **Conda list**

Setelah melakukan installing library **pandas** tahapan selanjutnya adalah melihat daftar library terinstall pada environtment kelasdata01 dengan cara mengetikan ( **conda list** )



Gambar 2.7 Anaconda Prompt(Anaconda3) **list library terinstall**

Gambar diatas merupakan sekumpulan library terinstall dan siap digunakan pada **environtment kelasdata01**

Pada tahapan berikutnya akan coba di demokan bagaimana caranya bekerja secara team dengan melakuan export dan create .yml.

**Lalu mengapa harus melakukan export dan create .yml?**

Export dan create .yml adalah dua perintah penting pada Anaconda Prompt yang digunakan untuk membuat dan mengelola environment.

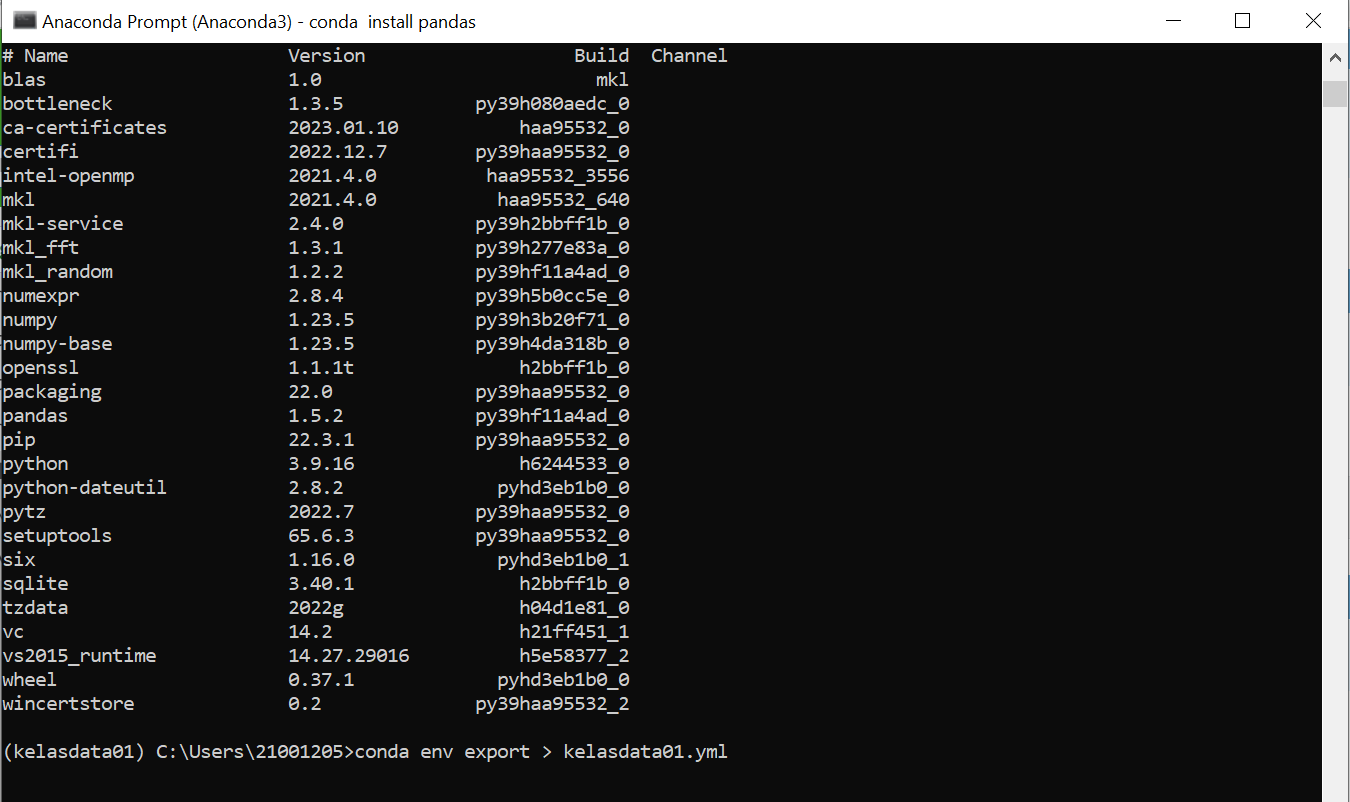
Ketika kita membuat environment baru di Anaconda, kita dapat menginstal package yang dibutuhkan dan mengonfigurasi lingkungan sesuai dengan kebutuhan proyek. Setelah itu, kita dapat melakukan eksport environment yang dibuat ke dalam file YAML (.yml) dengan perintah "conda env export > environment.yml". File YAML ini dapat dibagi dengan tim atau disimpan sebagai cadangan, dan dapat digunakan untuk mereplikasi environment yang sama di komputer lain.

Ketika kita ingin memulai proyek baru pada komputer lain atau dengan setup yang berbeda, kita dapat membuat environment baru menggunakan file YAML tersebut dengan perintah "conda env create -f environment.yml". Ini akan menginstal semua package yang diperlukan dan mengatur lingkungan seperti yang terdefinisi dalam file YAML.

**Fungsi** dari eksport dan create .yml pada Anaconda Prompt adalah untuk memastikan konsistensi lingkungan dan memastikan bahwa semua package dan dependensi yang diperlukan terinstal dengan benar. Ini juga dapat membantu dalam menangani masalah yang muncul saat bekerja dengan proyek dengan lingkungan yang kompleks atau ketika bekerja dengan tim.

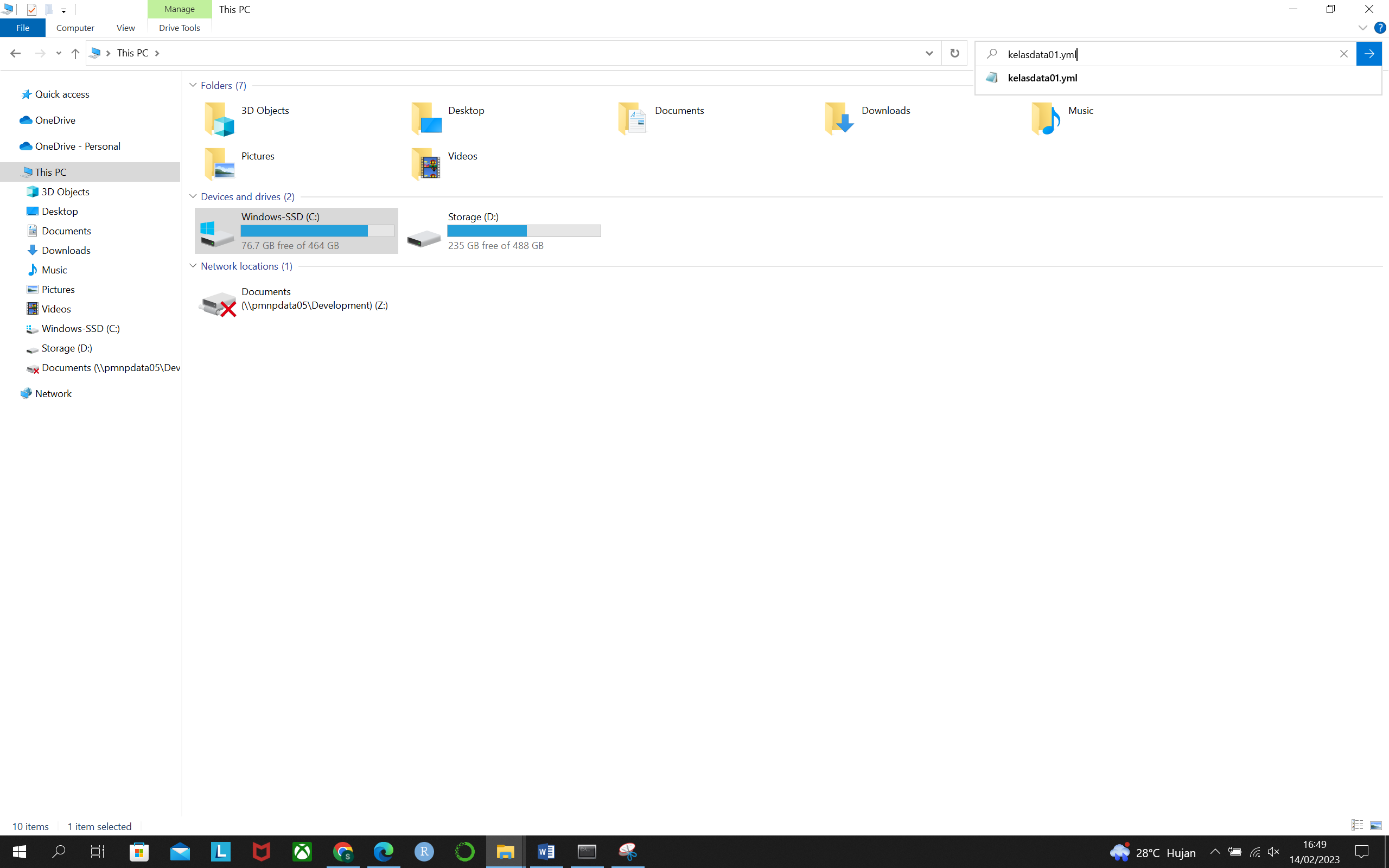
Dalam keseluruhan, eksport dan create .yml adalah alat yang berguna untuk mengelola environment pada Anaconda, dan memudahkan dalam memastikan konsistensi environment pada proyek tertentu.

**Berikut adalah langkah bagaimana caranya melakukan export dan create .yml**



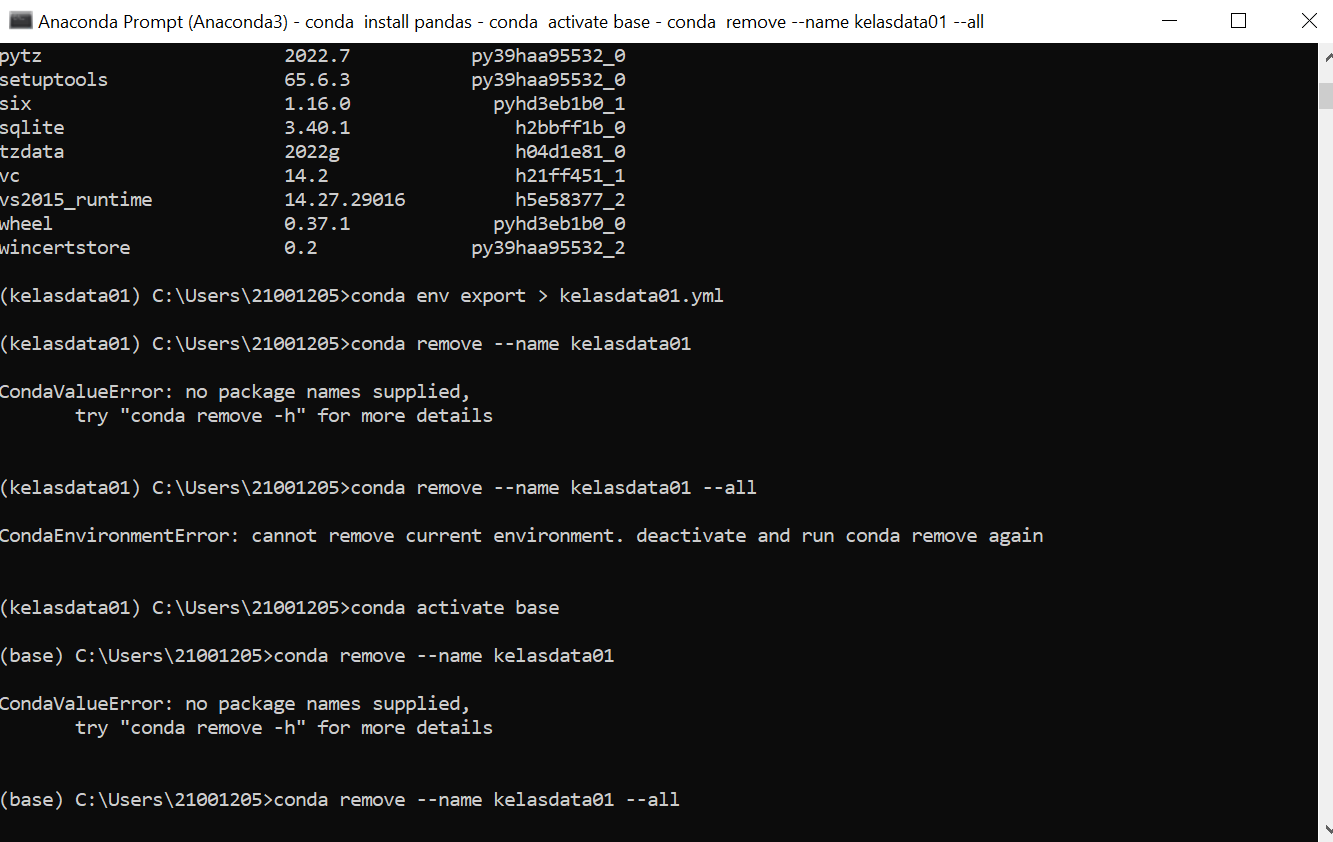
Gambar 2.8 Anaconda Prompt(Anaconda3) **export .yml**

Gambar diatas menunjukan cara melakukan export .yml ( **conda env export > kelasdata01.yml** ) dimana semua library dan package yang terinstall pada **environtment** **kelasdata01** akan diexport menjadi **.yml file**, cara mudah untuk menemukan file tersebut ada pada step selanjutnya.



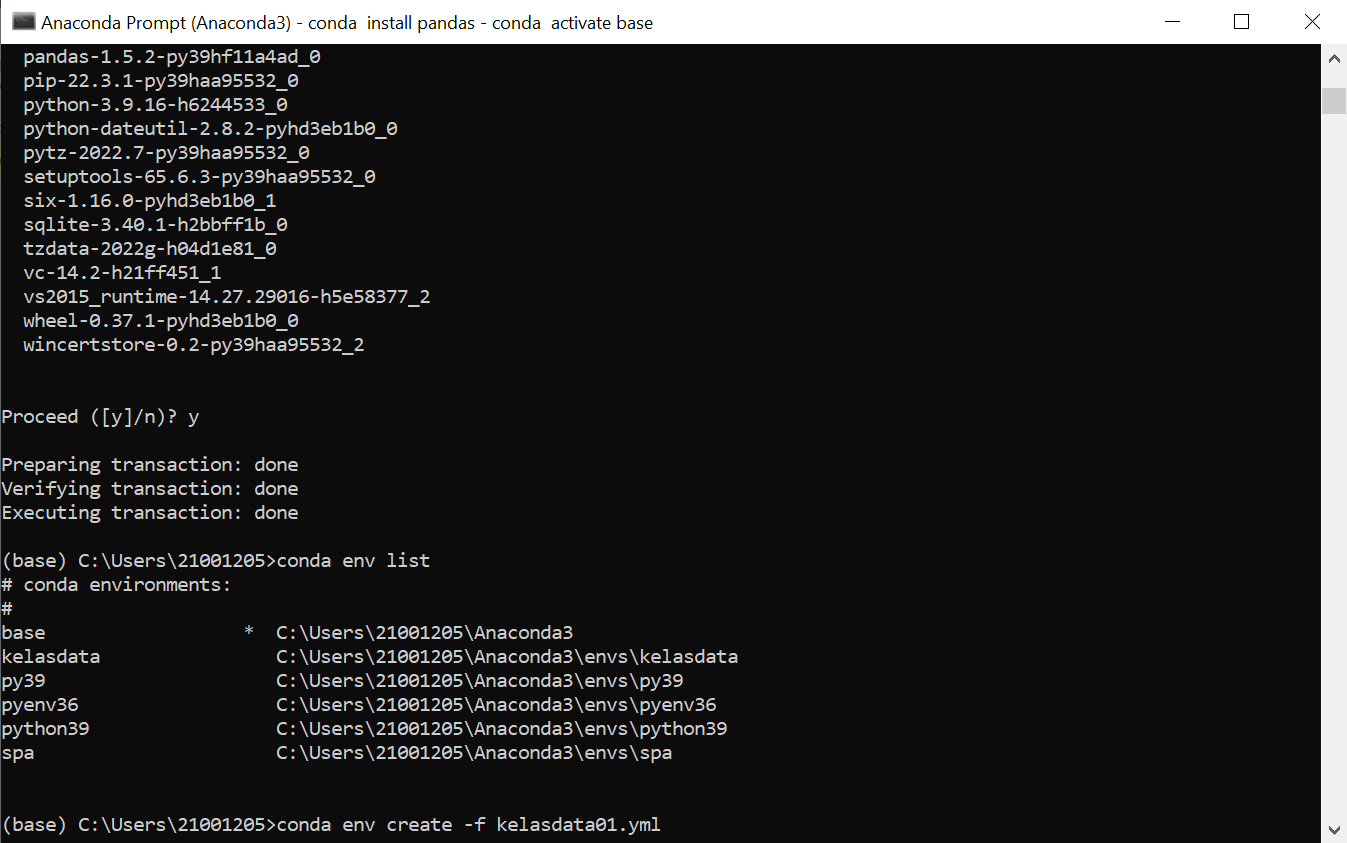
Gambar 2.9 This PC **kelasdata01 .yml**

Pada gambar 1.11 lakukanlah search pada this PC dengan mengetikan **kelasdata01.yml**, pada tahap sebelumnya kita menginisasikan nama .yml tersebut harus sama dengan nama environtmentnya ya.. agar mudah di cari, lalu pada tahapan selanjutnya akan mencoba melakukan remove atau menghapus environtment pada **kelasdata01** di **anaconda prompt**, dan melakukan import dari **kelasdata01.yml** yang sudah di eksport.



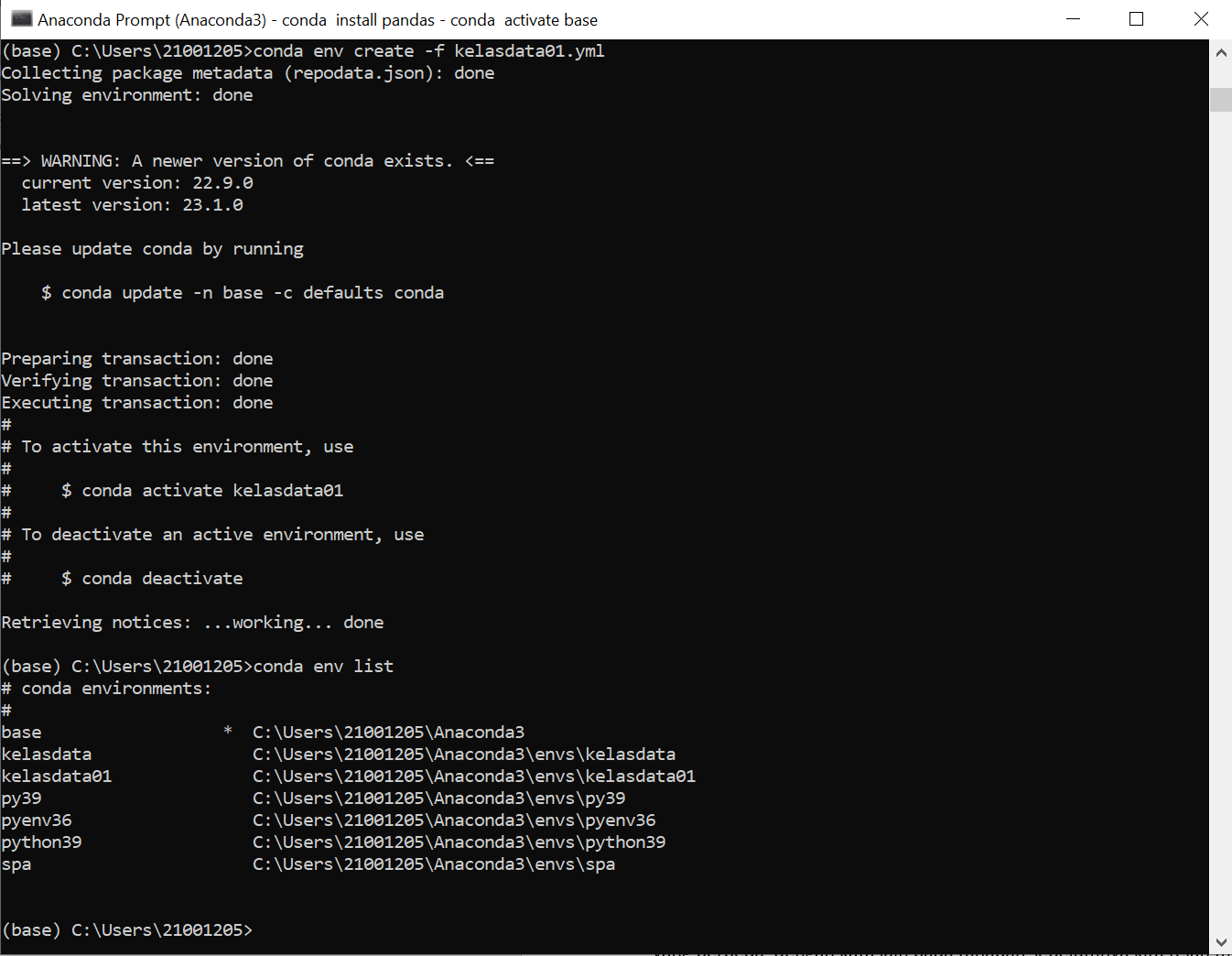
Gambar 2.10 This PC Anaconda Prompt(Anaconda3) **remove environtment**

Tahap pertama ketika ingin melakukan remove atau menghapus environtment anda harus berpindah environtment terlebih dahulu kedalam base, lalu mengetikan ( **conda remove --name kelasdata01 --all** ),dimana perlu diketahui setelah --name kita mengetikan nama environtment yang akan di hapus dalam kasus ini nama environtment yang akan dihapus adalah **kelasdata01.**



Gambar 2.11 This PC Anaconda Prompt(Anaconda3) check **removed environtment** dan **import kelasdata.yml**

Setelah melakukan remove pada environtment kelasdata01 kita dapat melihat pada gambar diatas bahwa environtment tersebut telah tiada dan akan dilakukan importing env yang sama dengan sumber yang berbeda, dengan kata lain pada tahapan sebelumnya kita telah mengetahui pentingnya export .yml pada environtment dan kita telah melakukan hal tersebut, pada scema ini environtment kita telah tiada dengan cara menyengaja melakukan deleted pada environtment anaconda promptnya, namun kita memiliki environtment yang telah di export sebelumnya, maka kita akan melakukan import environtment tersebut dengan cara mengetikan ( **conda env create -f kelasdata01.yml** ).



Gambar 2.12 This PC Anaconda Prompt(Anaconda3) check daftar **environtment**

Dapat terlihat dari gambar diatas bahwa anda telah berhasil mengimport kelasdata01.yml dan telah memiliki environtment tersebut kembali.

**Installing Notebook Extension**

**Jupyter Notebook** adalah lingkungan komputasi interaktif yang populer digunakan oleh banyak orang dalam berbagai bidang, seperti ilmu data, ilmu komputer, dan sains. Jupyter Notebook memungkinkan pengguna untuk membuat dan berbagi dokumen yang menggabungkan kode, teks, dan visualisasi. Kode dapat dieksekusi secara interaktif dan hasilnya ditampilkan langsung dalam dokumen yang sama.

Meskipun Jupyter Notebook sudah sangat berguna, Anda dapat memperluas fungsionalitasnya dengan menginstal Notebook Extension. Notebook Extension adalah paket perangkat lunak yang dapat diinstal pada Jupyter Notebook untuk menambahkan fitur tambahan, fungsionalitas, dan integrasi dengan alat-alat lain.

Salah satu manfaat utama dari menginstal **Notebook Extension** adalah untuk menambahkan fitur tambahan. Misalnya, extension seperti Code Folding dapat membantu pengguna untuk mengorganisir kode yang lebih panjang dan memungkinkan pengguna untuk menampilkan atau menyembunyikan blok kode tertentu. Extension lain seperti Table of Contents (TOC) membuat daftar isi dokumen jupyter notebook secara otomatis, sehingga memudahkan pengguna untuk menavigasi dokumen yang panjang.

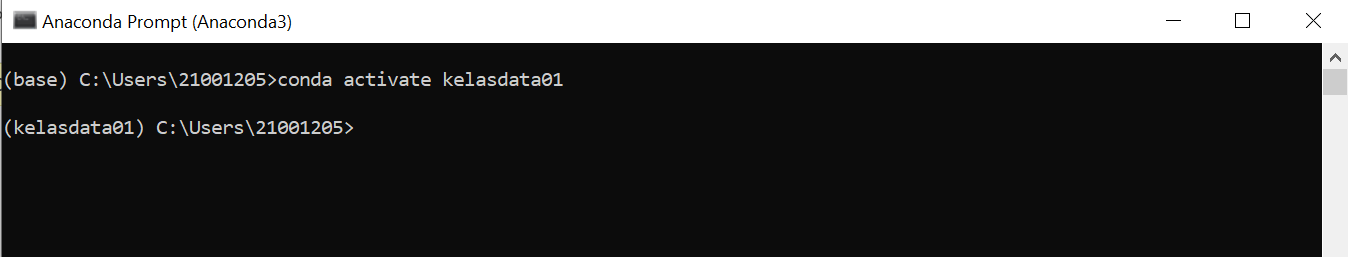
Menginstal Notebook Extension juga dapat meningkatkan produktivitas. Dengan menginstal extension yang tepat, pengguna dapat mempercepat proses pengembangan dan analisis data. Misalnya, extension seperti Hinterland dapat memberikan saran otomatis saat mengetik kode, sehingga pengguna dapat menghemat waktu dan meminimalkan kesalahan.

Beberapa Notebook Extension juga menyediakan integrasi dengan alat lain seperti Git, Conda, dan Docker. Contohnya, Git Extension dapat membantu pengguna dalam mengelola proyek di repositori Git, sementara Docker Extension memudahkan pengguna untuk menjalankan lingkungan Jupyter Notebook dalam kontainer Docker.

Menginstal Notebook Extension juga memungkinkan pengguna untuk menyesuaikan tampilan Jupyter Notebook. Misalnya, dengan extension seperti Jupyter Themes, pengguna dapat mengubah tema Jupyter Notebook sesuai dengan preferensi mereka, dan membuat lingkungan Jupyter Notebook mereka lebih mudah dilihat dan digunakan.

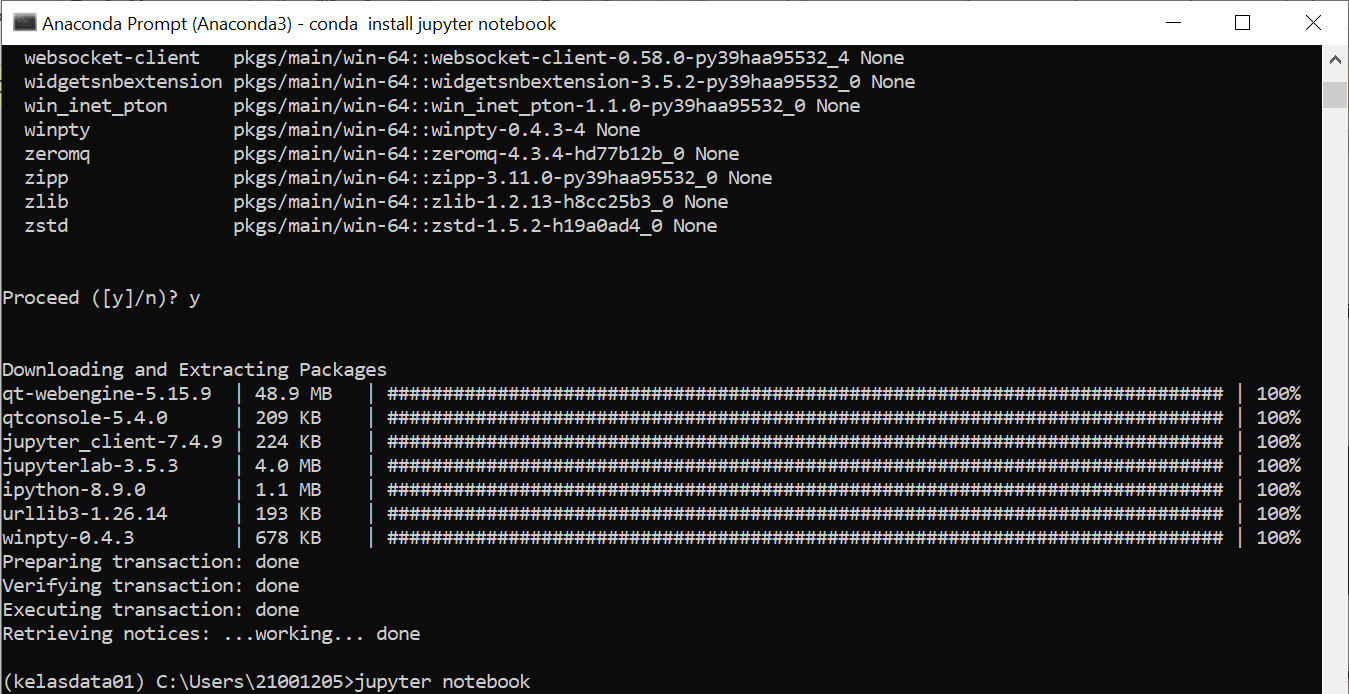
Terakhir, menginstal Notebook Extension memungkinkan pengguna untuk mengembangkan dan mengeksperimen dengan ide-ide baru untuk meningkatkan fungsionalitas dari Jupyter Notebook. Dengan mengembangkan extension sendiri, pengguna dapat menyesuaikan lingkungan Jupyter Notebook mereka sendiri, yang akan memungkinkan mereka lebih produktif dan efisien dalam bekerja.

Pada tahap selanjutnya disini akan dicoba melakukan instalasi notebook extension, dengan tujuan membuat jupyternotebook yang nantinya akan digunakan lebih interactive.



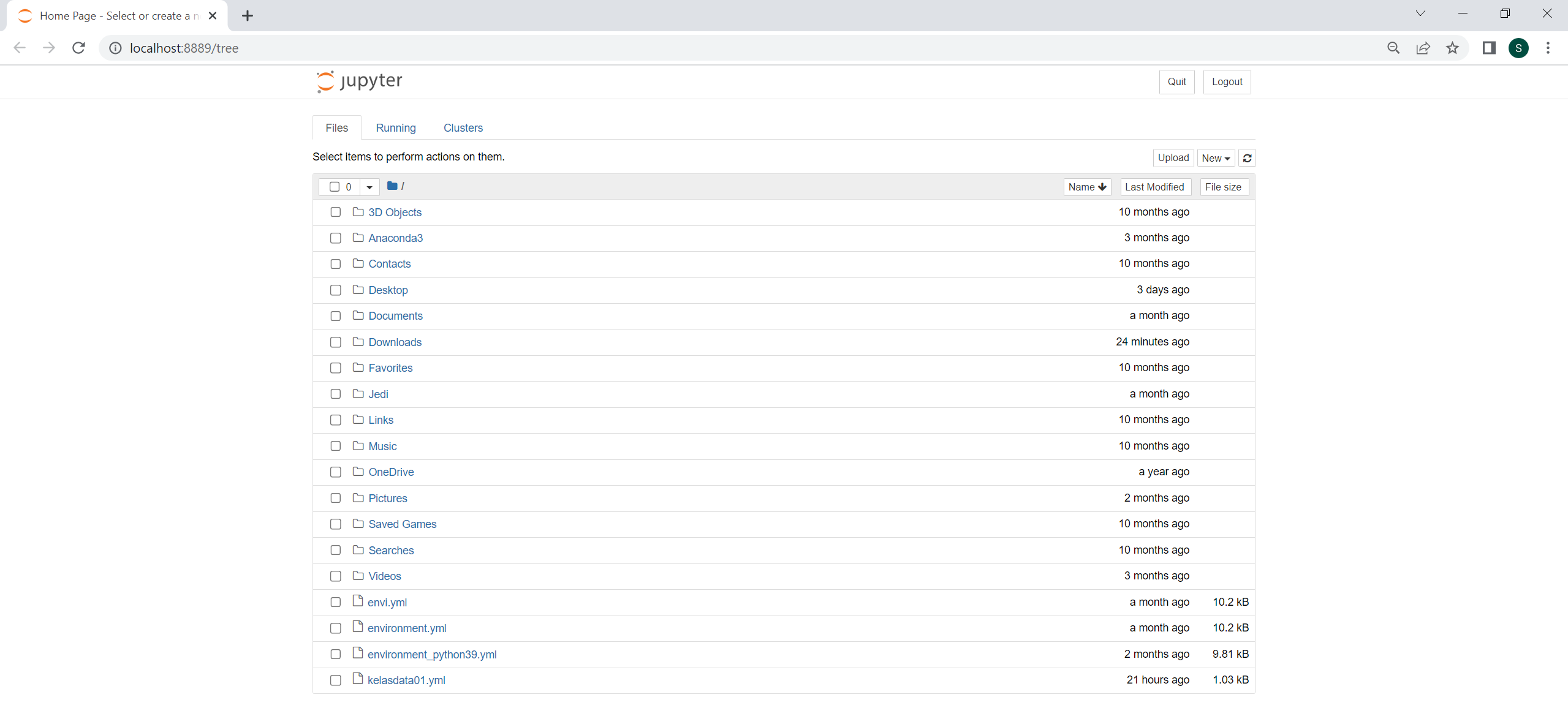
Gambar 3.0 This PC Anaconda Prompt(Anaconda3) masuk kedalam environtment **kelasdata01**

Masuk kedalam envorontment kelasdata01 dengan mengaktifkan **conda activate kelasdata01** lalu setelah masuk kedalam environtmentnya, berikan perintah dengan mengketikan **conda install jupyter notebook**.



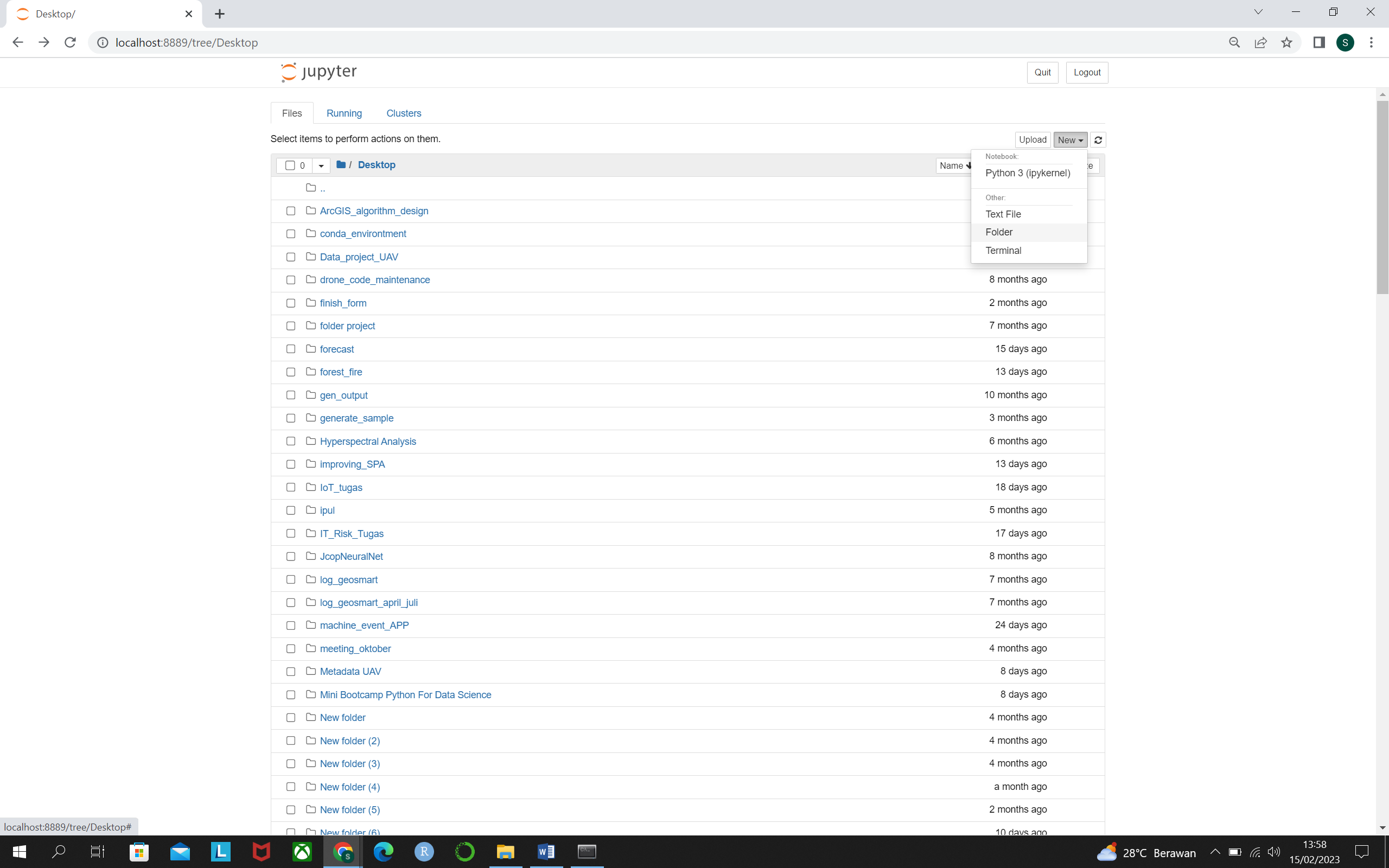
Gambar 3.1 This PC Anaconda Prompt(Anaconda3) masuk kedalam **environtment** **kelasdata01**

Setelah berhasil menginstall **jupyter notebook** pada **environtment** **kelasdata01** maka langkah selanjutnya kita akan masuk kedalam jupyter notebook pada **environtment** **kelasdata01**, dengan mengketikan **jupyter notebook** saja.



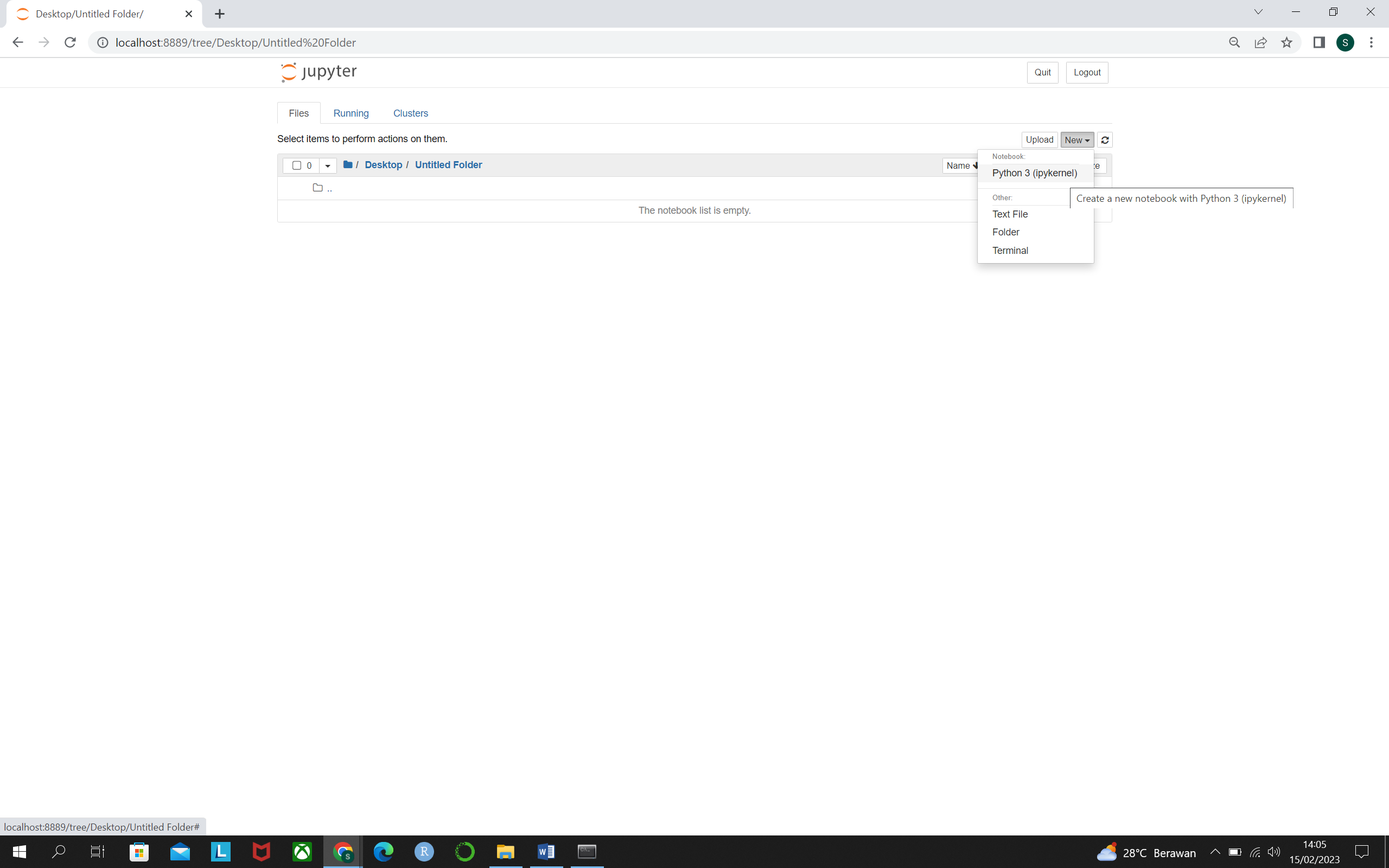
Gambar 3.4 Tampilan awal Jupyter Notebook

Setelah masuk kedalam tampilan awal jupyter notebook, tahap selanjutnya kita akan coba membuat folder baru dengan masuk kedalam desktop terlebih dahulu lalu masuk ke menu **New** pada jupyter notebook.



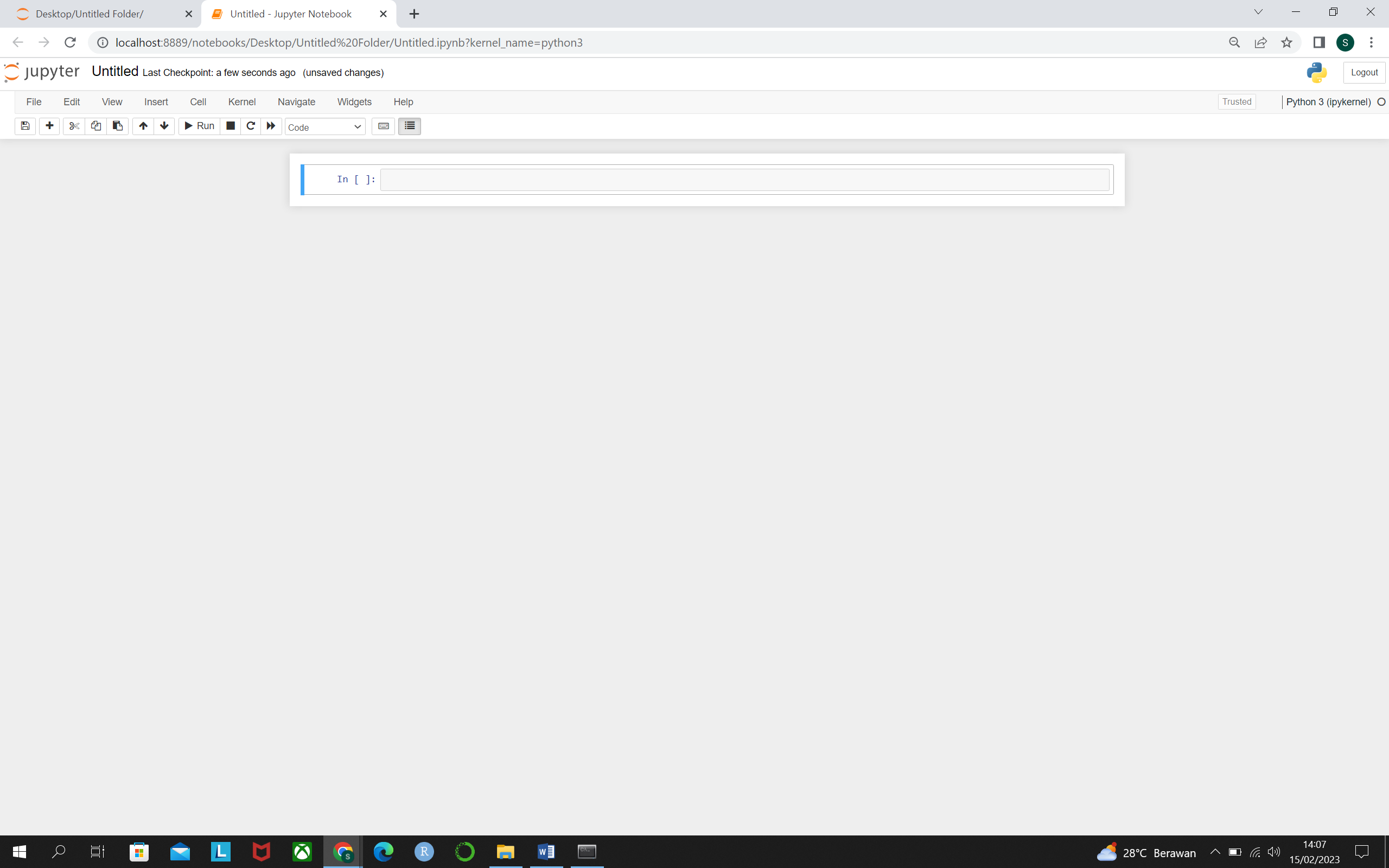
Gambar 3.5 Tampilan awal Jupyter Notebook

Setelah membuat folder baru pada desktop melalui jupyter notebook, maka folder akan otomatis terbuat dengan nama **Untiteld Folder** , masuk kedalam folder tersebut dan lihat tampilan awal pada IDE Jupyter notebook tersebut.



Gambar 3.6 Tampilan awal Jupyter Notebook

Membuat project baru pada folder yang telah di buat sebelumnya dengan memilih menu New dan pilih Python 3 (ipykernel).



Gambar 3.7 Tampilan awal Jupyter Notebook

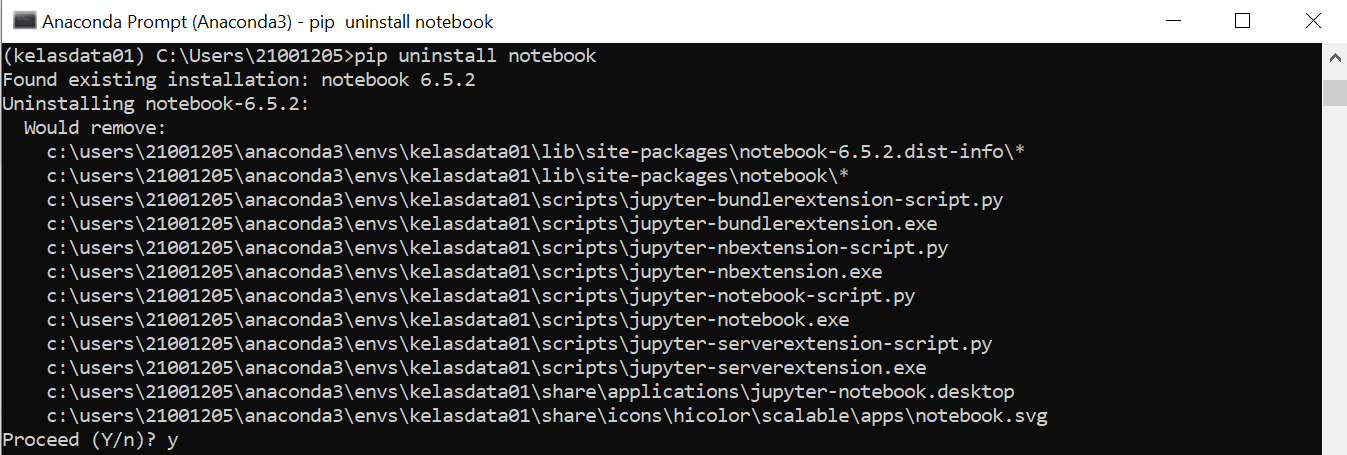
Berikut merupakan tampilan awal dalam Jupyter Notebook , perlu jadi catatan bahwa pada environtment kelasdata01 ini kita belum melakukan install notebook extension dan tampilannya hanya seperti ini saja, kurang begitu interactive, maka tahapan selanjutnya kita akan **melakukan install notebook extension** pada Anaconda Prompt di **environtment** **kelasdata01**.



Gambar 3.8 default notebook version

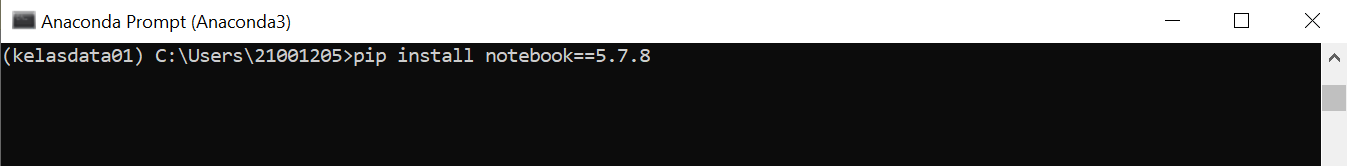
Perlu jadi catatan bahwa ketika kita melihat daftar library atau packages yang sudah kita install pada **environtment kelasdata01** tadi, di versi python 3.9 default dari notebook nya berada pada versi 6.5.2, sedangkan banyak extensi dari notebook yang hanya compatible di bawah itu, seperti yang ada di forum

<https://stackoverflow.com/questions/58146683/all-jupyter-contrib-nbextensions-are-marked-as-possibly-incompatible> dan <https://stackoverflow.com/questions/43916440/error-loading-jupyter-notebook-extensions> maka dari itu kita akan melakukan uninstall pada notebook di env kelasdata01, dan menginstall nya kembali dengan versi yang lebih rendah.



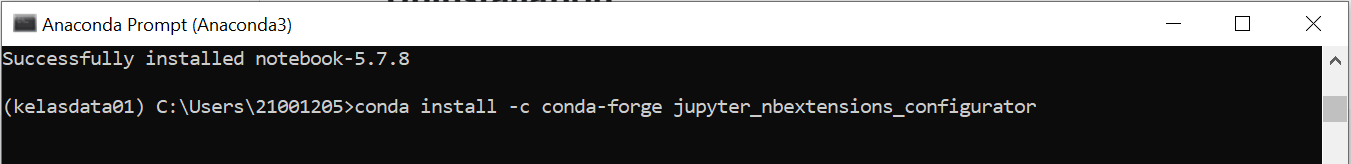
Gambar 3.9 uninstall notebook version > 6

Tahap pertama setelah mengaktivasi environtment kelasdata01, ketikan **pip uninstall notebook** dan pilih (y).



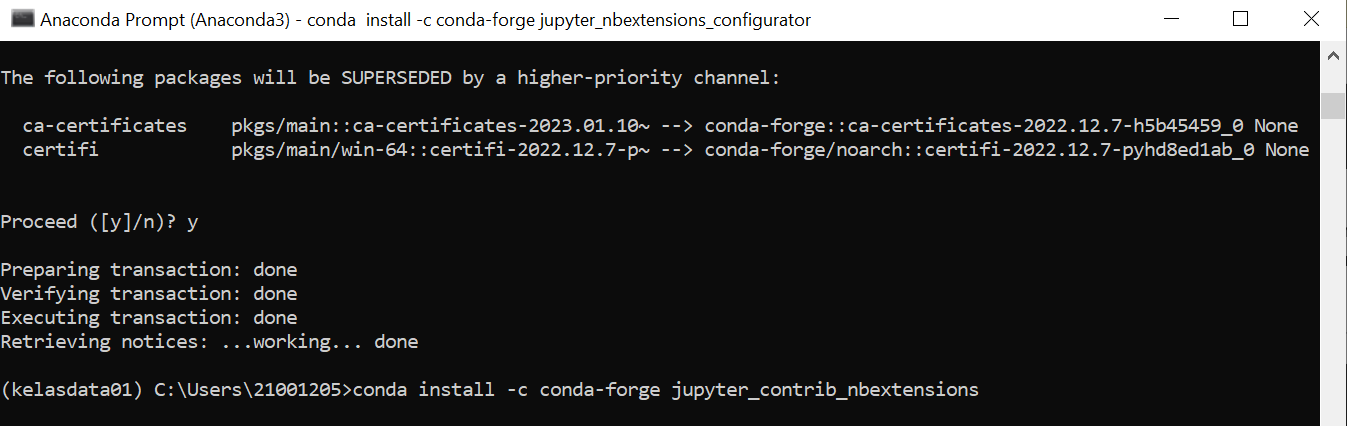
Gambar 3.10 install notebook version < 6

Tahap selanjutnya akan dilakukan install untuk notebook dengan versi dibawah 6, dalam hal ini saya memilih menginstall notebook 5.7.8 sesuai dengan link stackoverflow diatas dengan mengetikan **pip install notebook==5.7.8**



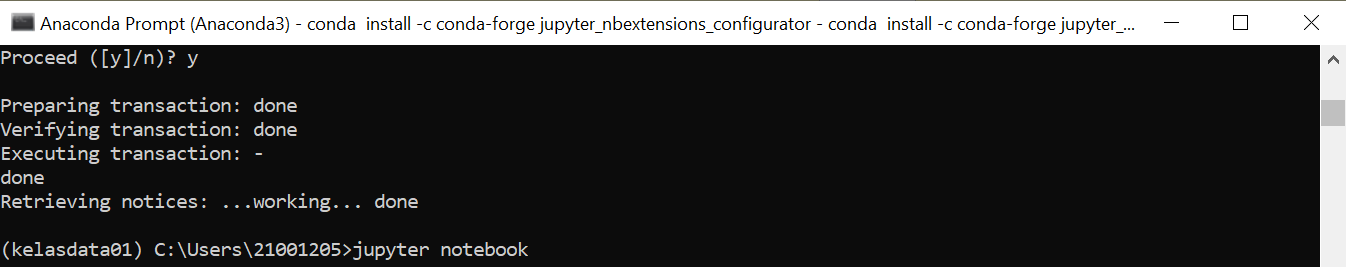
Gambar 3.11 install notebook nbextensions\_configurator

Setelah berhasil install notebook dengan versi dibawah 6 maka tahapan selanjutnya kita akan mencoba melakukan nbextension configuratornya terlebih dahulu dengan mengetikan **conda install -c conda-forge jupyter\_nbextensions\_configurator**



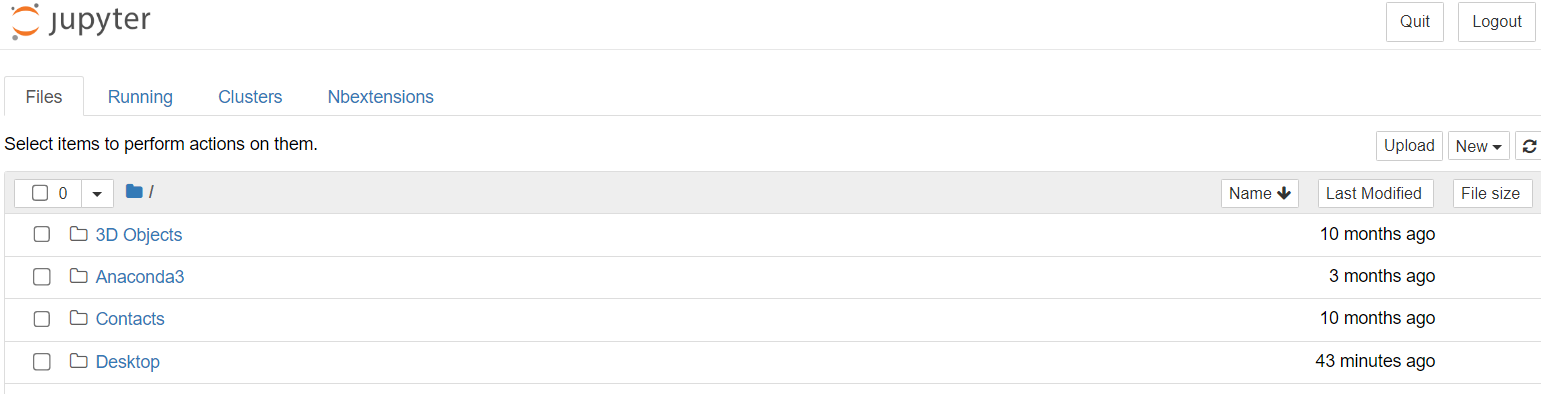
Gambar 3.12 install notebook jupyter\_contrib\_nbextensions

Setelah berhasil install nbextensions\_configurator maka tahapan akhirnya kita akan mencoba melakukan install nbextension contrib nya dengan **mengetikan conda install -c conda-forge jupyter\_contrib\_nbextensions**



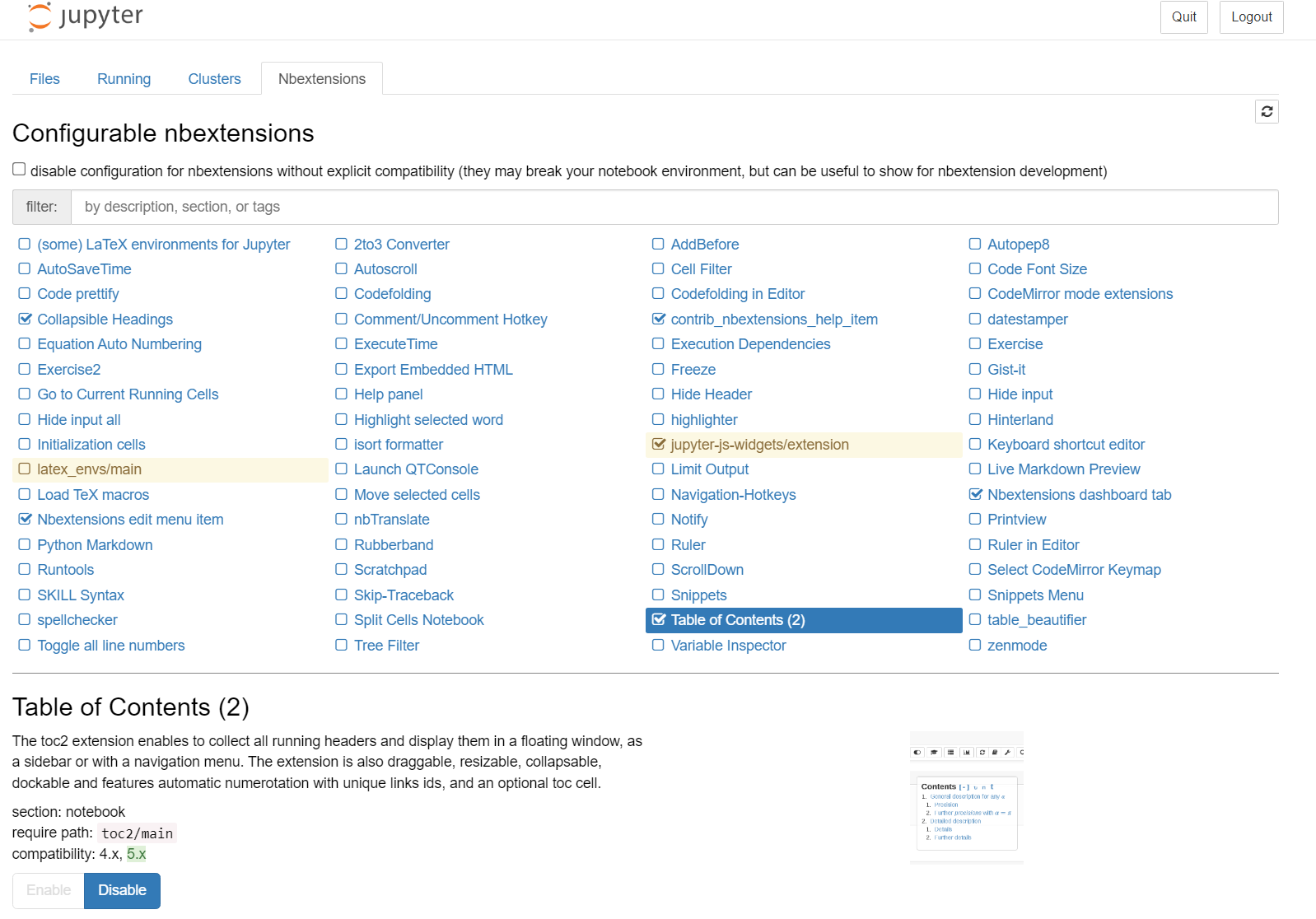
Gambar 3.13 install notebook jupyter\_contrib\_nbextensions

Setelah semuanya terinstal maka tahapan akhir dari sini kita akn mencoba melihat tampilan awal jupyter notebook nya lagi dengan **mengetikan jupyter notebook** pada env **kelasdata01**

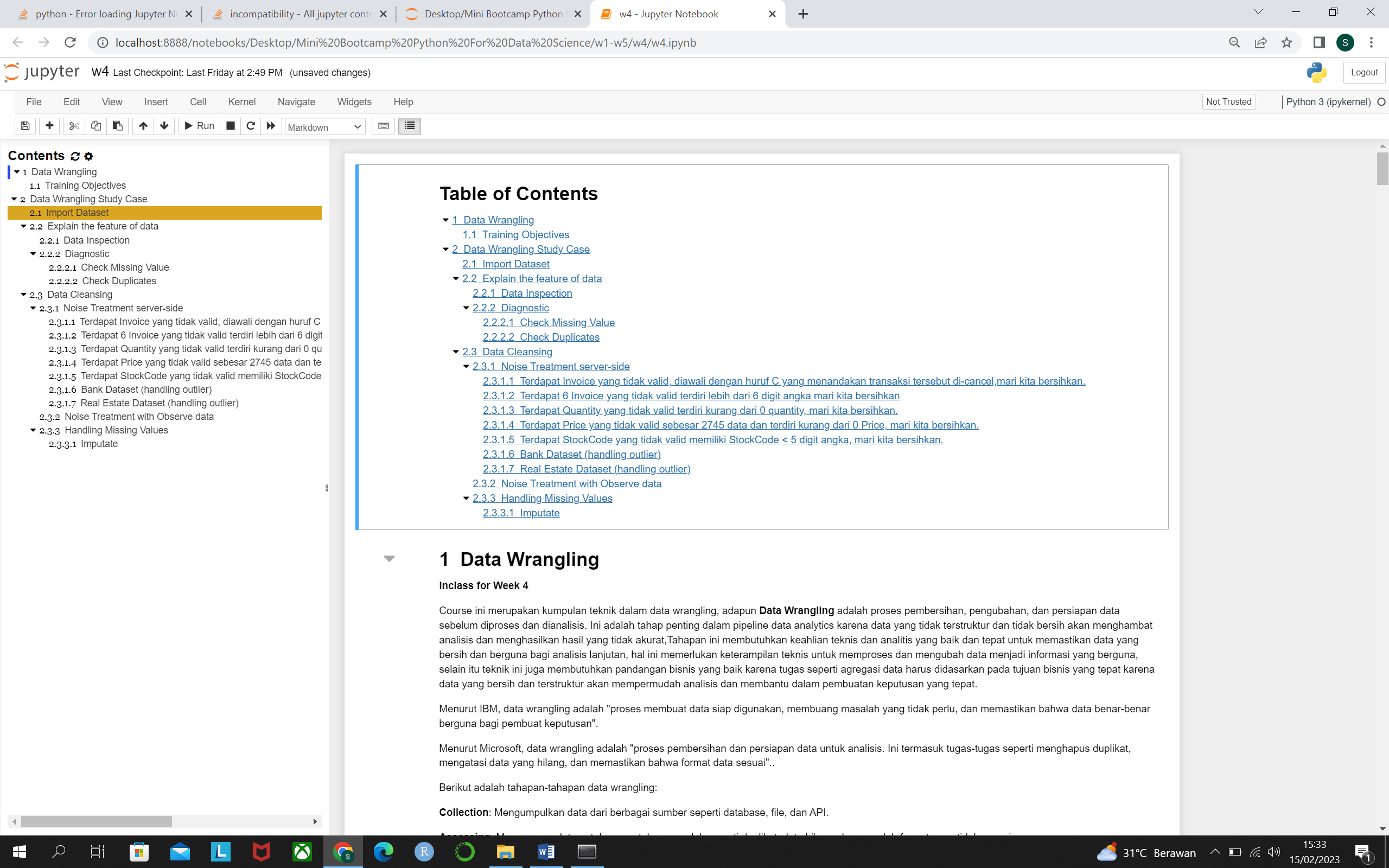


Gambar 3.14 berhasil extensi tambahan pada jupyter notebook

Dan Yaa kita telah berhasil menambahkan NoteBookextension pada Jupyter Notebook kita, yang menandakan bahwa tahapan akhir dari instalasi NBextension telah selesai namun bukan tahapan akhir dari course ini, tahapan akhir nya ada pada setting extensi yang dibutuhkan pada jupyternotebook, dan dalam course ini hanya di contohkan menggunakan extensi Table Of Content dengan tujuan menjadikan Jupyter Notebook jauh lebih interactive lagi.

 Gambar 3.15 berhasil extensi tambahan pada jupyter notebook

Masuk kedalam NBextensions lalu Enable Table Of Content dengan menceklis nya, pada tahapan ini deskripsi dari setiap extensi akan terlihat dibagian bawahnya, sebagai contoh disini hanya mengaktifkan Table Of Content saja.



Gambar 3.16 Jupyter Notebook with Table Of Content

Gambar diatas menunjukan bahwa kita telah berhasil melalui semua tahapan install nbextensions pada jupyter notebook, hasil diatas terlihat jupyter notebook lebih interactive dari sebelumnya, adapun jika belum berhasil dan tidak terlihat content pada side bar nya maka langkah yang harus dilakukan pada Anaconda Prompt nya adalah sebagai berikut:

* conda remove --force jupyter notebook
* conda remove --force jupyter\_nbextensions\_configurator
* conda remove --force jupyter\_contrib\_nbextensions
* conda install jupyter notebook
* pip uninstall notebook
* pip install notebook ==5.7.8
* conda install -c conda-forge jupyter\_nbextensions\_configurator
* conda install -c conda-forge jupyter\_contrib\_nbextensions

HandBook ini saya buat dari berbagai refrensi dan dokumentasi di internet yang telah di cantumkan pada isi content tersebut, adapun versi python yang dijalankan pada handbook ini yaitu versi python3.9

Best Regards,

Sandy Putra Utama