



Society of Petroleum Engineers

**SPE-217148-MS**

## **Introducing Python Coding to Petroleum Engineering Undergraduates: Excerpts from a Teaching Experience**

O. O. Mosobalaje and O. D. Orodu, Petroleum Engineering Department, Covenant University, Ota, Ogun State, Nigeria

Copyright 2023, Society of Petroleum Engineers DOI [10.2118/217148-MS](https://doi.org/10.2118/217148-MS)

This paper was prepared for presentation at the SPE Nigeria Annual International Conference and Exhibition held in Lagos, Nigeria 31 July - 2 August 2023.

This paper was selected for presentation by an SPE program committee following review of information contained in an abstract submitted by the author(s). Contents of the paper have not been reviewed by the Society of Petroleum Engineers and are subject to correction by the author(s). The material does not necessarily reflect any position of the Society of Petroleum Engineers, its officers, or members. Electronic reproduction, distribution, or storage of any part of this paper without the written consent of the Society of Petroleum Engineers is prohibited. Permission to reproduce in print is restricted to an abstract of not more than 300 words; illustrations may not be copied. The abstract must contain conspicuous acknowledgment of SPE copyright.

---

### **Abstract**

The post-Covid world is witnessing a rise in automation and remote work models. Oilfield operations are becoming increasingly innovation-driven with advances such as digitalization technologies, smart fields and intelligent wells. Proliferation of data is extending career frontiers in data analytics, machine learning and artificial intelligence. Human competence in computer programming is a key enabler of these trends. As a contribution to the Nigerian oil/gas human resources development, the petroleum engineering program at Covenant University recently developed and is implementing a course module on Python programming with oil/gas applications. This paper documents the philosophy, pedagogy, and prospects of this initiative and provides a guide for its implementation across the Nigerian educational space.

The module opens with a seminar on the emerging oil/gas opportunities in data science – to stimulate students’ interest. Thereafter, a gentle introduction to computer programming is taught. At its core, the module teaches basics of Python programming language – input/output, objects (values, variables, keywords), conditional and repetitive structures, *functions*, *lists*, *tuples* and *dictionaries*. The module is enriched with applications in reservoir volumetrics, material balance equation, PVT properties, reservoir discretization and simulation. Hands-on experience is enhanced with class demos and take-home programming assignments featuring simple algorithms. Also, the course features a training on the use of distributed version control (*GitHub*) for collaboration between students and instructors. All course materials are available on an open-access *GitHub* repository, with hyperlinks embedded in lecture notes. Ultimately, the course assesses students’ skills with exams set in the context of quasi-real-life projects. The future prospects targeted in this initiative includes a follow-up module on petroleum data analytics and machine learning, incorporation of Python coding into other modules, and a short-course for industry professionals.

### **Introduction**

#### **The Motivation**

In the face of more oilfields getting matured, new discoveries occurring less frequently, and competition from non-fossil sources; oilfield operations are becoming increasingly innovation-driven. Digitalization

technologies, smart fields and intelligent wells are some of the recent advances in oilfield operations. These advances are enabled by technologies such as cloud computing, Internet of Things (IoT), Artificial Intelligence (AI). Moreover, the post-Covid world is witnessing a rise in labour automation and remote work models. Also, the proliferation of oilfield data acquisition technologies is extending career frontiers in petroleum data analytics and machine learning (PDA&ML). Recently, IBM reported that about 80,000 sensors generate data on an average offshore platform (IBM, 2020). Increasingly, computer systems are becoming capable of intelligently managing oilfields (Du *et al.*, 2020). Mathieson, Meehan and Potts (2019) identified productivity improvements, deliverable only by AI and digitalization, as part of the issues confronting the next generation of petroleum engineers. Unarguably, human competence in computer programming (coding) is crucial in driving these innovations. In recognition of changes in the industry, Feder (2019) advocates for a petroleum engineering education that produces graduates with both domain and digital knowledge. In advocating for the incorporation of industry-based oil/gas softwares into petroleum academic curriculums, Elochukwu, Joukarborazjany and Attarhamed (2013) recognized that introducing computer programming to petroleum engineering undergraduates is a necessary pre-requisite. Notably, the Society of Petroleum Engineers (SPE), in a recent update to its Competency matrices, included "analytics programming language" in the data science and engineering analytics matrix (SPE, 2021). It therefore becomes imperative for petroleum engineering (PE) undergraduate programs to equip upcoming petroleum engineers with computer coding skills required to fit into the emerging opportunities. It is in response to these issues that the petroleum engineering program at Covenant University recently developed and is implementing a course module on Python programing with petroleum engineering applications. This initiative is part of the University's contribution to the Nigerian oil/gas human resources development, and a demonstration of its commitment to United Nation's SDGs 4, 7, 8 and 9. This paper presents a documentation of the initiative – its philosophy, pedagogy, performance and prospects. It is motivated by the need to recommend and provide a working guide for the adoption and implementation of similar initiatives across the petroleum engineering educational space in Nigeria. Additionally, we seek feedback suggestions from industry professionals and fellow academic instructors on the content of the course module as presented in this paper.

In planning and implementing the course module, we adopted the *Outcome-based Education* (OBE) approach. The OBE approach is predicated on the need to equip students with knowledge, competencies (skills) and qualities needed to be successful upon graduation (Quadri *et al.*, 2020). Emerging workplace trends are motivating academic programs across the world to build curricula with strong emphasis on skills acquisition. In this paper, we specifically deployed the *Understanding by Design* (UbD) Framework (Wiggins and McTighe, 2005). Skills acquisition is a foremost desired outcome in the UbD Framework.

### The Choice of Python

The initiative to incorporate the Python programming course module into the petroleum engineering program was adopted at a recent curriculum review at CU. Two computer application course modules featured in the previous curriculum prior to the review. Those modules featured softwares such as legacy FORTRAN, MATLAB® and Microsoft Excel®. However, the reviewed curriculum features a single module with Python as the choice programming language. The switch to Python, a general-purpose programming language, is predicated on a number of considerations such as popularity, license, user-support, extensiveness, ease of learning, user-friendliness, and readability. As of March, 2023, Python tops the list of Popularity of Programming Language (PYPL) index (Figure 1), as per data curated from Google Trends (PYPL, 2023). Another strong consideration in favour of Python is its open-source license which implies it is freely available for use and re-distribution, even for commercial purposes (Python Software Foundation, 2023). Python's open-source status makes it accommodates thousands of third-party modules contributed by the users' community leading to a very extensive functionalities and features. Its popularity also implies a very robust support base driven by the vast users' communitiy. Python code syntax, being

rather similar to the English language, makes it easy to learn for beginners and enhances its readability. It is our considered opinion that none of the softwares in the previous curriculum combines such benefits as Python does; hence, the switch to Python in the reviewed curriculum.

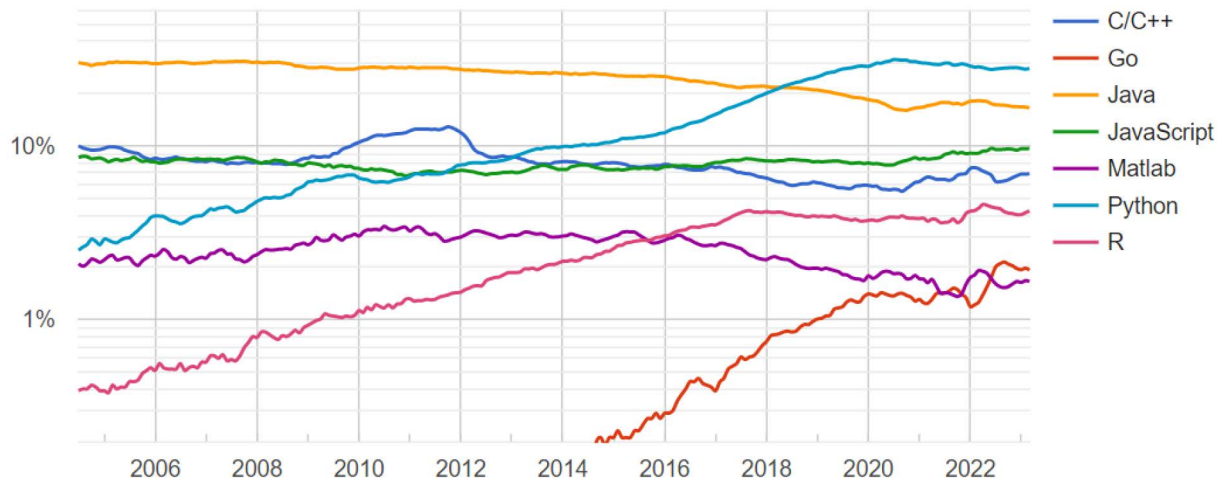


Figure 1—Popularity of Programming Language (PYPL) Index (Source: PYPL, 2023)

## The Course Module Design: Content, Assessment and Pedagogy

In this paper, we opted to present the course module design using the UbD Framework (Wiggins and McTighe, 2005), in keeping to terms with the OBE approach. Essentially, the UbD Framework (also known as backward design) starts curriculum design with the desired learning outcomes (Content) and plans backward by setting evidence of achievement of outcomes (Assessment) and instructional methods and strategies (Pedagogy).

### Content: Desired Learning Outcome

Here, we present the intended skills: actionable things the students should be able to do at the end of the course module. The popular Bloom's Taxonomy verbs, as revised by Anderson and Krathwohl (2001) were deployed in communicating these learning outcomes. On the cognitive dimension of the taxonomy, most of the learning outcomes are at the 'apply' and 'create' levels of the pyramid. Also, most of the outcomes are *conceptual* and *procedural* on the knowledge dimension. The learning outcomes are presented hereunder.

At the end of this course, students should be able to:

1. Analyze the workings of fundamental patterns in Python codes such as input/output statements, execution flow, control structures etc.
2. Create and assign values to variables; convert values/variables from one type to another; write executable Python statements involving mathematical and string operators.
3. Develop Python scripts to implement conditional execution (*if...else*), nested and chained *if...else* and repetitive (*for* and *while*) loops.
4. Create and call custom functions for common PE computational tasks; set or skip values for positional or keyworded arguments of functions.
5. Create and (re-)assign multiple values to data structures: *lists*, *tuples* and *dictionaries*; access element(s) of data structures; loop through the indices or values in data structures.
6. Automate common petroleum engineering computational tasks with Python code scripting.
7. Develop algorithms and Python scripts to execute petroleum engineering computational workflows.

## Assessment: Evidence of Outcome Attainment

Consequent on setting the desired learning outcomes, the definite assessment rubrics are here presented in Table 1. These assessment rubrics serve as evidences that a certain student have attained the desired outcome. In essence, these are specific items by which students' performance in the course are evaluated. An essential element of the Ubd Framework is the alignment of the desired outcomes with the assessment rubrics (Wiggins and McTighe, 2005). In keeping true to the framework, we mapped the various learning outcomes to specific assessment rubrics thereby ensuring all desired outcomes are adequately assessed. Also, specific PE examples are presented with the rubrics.

Table 1—Assessment Rubrics

Learning Outcome (LO)	Assessment
LO1: Students should be able to analyze the workings of fundamental patterns in Python programs such as input/output statements, execution flow, control structures etc.	<p><b>Acceptable evidence</b> of this learning outcome will be:</p> <ul style="list-style-type: none"> <li>Ability of the students to construct Python statements to request input parameters from program users.</li> <li><b>Example:</b> porosity for use in a volumetric program</li> <li>Ability of the students to construct Python statements to report the output of a computation/simulation.</li> <li><b>Example:</b> cumulative oil produced, material balance simulator program.</li> </ul>
LO2: Students should be able to create and assign values to variables; convert values/variables from one type to another; write executable Python statements involving mathematical and string operators.	<p><b>Acceptable evidence</b> of this learning outcome will be:</p> <ul style="list-style-type: none"> <li>Ability of the students to create variables to hold computation parameters and outputs of a kinds (numeric, string, categorical or Boolean etc).</li> <li><b>Example:</b> a variable to hold lithology types ('shale', 'sandstone', 'limestone')</li> <li>Ability of the students to construct Python statements to convert values from one type to another.</li> <li><b>Example:</b> conversion of user-input from string to numeric before computation.</li> <li>Ability of the students to implement petroleum engineering equations/formulae in Python programs without missing the order of operations (BODMAS).</li> <li><b>Example:</b> cost-per-foot formula in drilling engineering.</li> </ul>
LO3: Students should be able to develop Python scripts to implement conditional execution ( <i>if...else</i> ), nested and chained <i>if...else</i> and repetitive ( <i>for</i> and <i>while</i> ) loops.	<p><b>Acceptable Evidence</b> of this learning objective will be:</p> <ul style="list-style-type: none"> <li>Ability of students to identify binary courses in workflows, formulate Boolean expressions and construct Python <i>if</i> or <i>if...else</i> statements</li> <li><b>Examples:</b> advancing or terminating simulation cycles depending of average reservoir pressure value; Sutton's gas gravity correction depending on impurity levels.</li> <li>Ability of students to formulate a <i>for</i> loop to implement definite repetitive tasks in a workflow.</li> <li><b>Example:</b> looping through gridblocks and computing flow parameters in a discretized reservoir model.</li> <li>Ability of students to formulate a <i>while</i> loop to implement indefinite repetitive tasks in a workflow.</li> <li><b>Example:</b> updating oil formation volume factor (FVF) at series of decreasing pressure values until bubble point pressure is attained.</li> </ul>
LO4: Students should be able to create and call custom functions for common PE computational tasks; set or skip values for positional or keyworded arguments of functions.	<p><b>Acceptable evidence</b> of this learning objective will be:</p> <ul style="list-style-type: none"> <li>Ability of students to construct the <i>header</i> of a function, listing relevant arguments.</li> <li><b>Example:</b> a function to compute solution gas-oil ratio, <math>R_s</math>.</li> <li>Ability of students to specify default values for function arguments.</li> <li><b>Example:</b> setting standard temperature and pressure (STP) values in natural gas density computations</li> </ul>

Learning Outcome (LO)	Assessment
	<ul style="list-style-type: none"> <li>• Ability of students to pass a function's output value via a <i>return</i> statement.</li> <li>• <b>Example:</b> returning flowrate, <i>q</i>, from a function written to implement Vogel's inflow performance relationship</li> <li>• Ability of students to alternate between positional and keyworded argument passing when calling in-built or custom functions.</li> </ul>
LO5: Students should be able to create and (re-)assign multiple values to data structures: <i>lists</i> , <i>tuples</i> and <i>dictionaries</i> ; access element(s) of data structures; loop through the indices or values in data structures.	<p><b>Acceptable evidence</b> of this learning objective will be:</p> <ul style="list-style-type: none"> <li>• Ability of students to deploy Python's in-built data structures (<i>lists</i>, <i>tuples</i> and <i>dictionaries</i>) in storing data multi-valued data encountered in PE workflow.</li> <li>• <b>Example:</b> storing gridblock permeability values to be used in a reservoir flow simulator.</li> <li>• Ability of students to extract values stored in multi-valued data structures for use in repetitive workflows.</li> <li>• <b>Example:</b> extracting gridblock pressure for use in material balance computations</li> <li>• Ability of students to use a <i>for... loop</i> to iterate through a multi-valued data structure.</li> <li>• Ability of students to match the 1-D indices of the data structures with the 2-D gridblock indices in a discretized reservoir.</li> <li>• Ability of students to modify data structures using various in-built <i>methods</i> and <i>functions</i>.</li> <li>• <b>Example:</b> appending latest computed flowrate to a pre-defined <i>list</i>, <i>tuple</i> or <i>dictionary</i> while looping through simulation time-cycles.</li> </ul>
LO6: Students should be able to automate common petroleum engineering computational tasks with Python code scripting.	<p><b>Acceptable evidence</b> of this learning objective will be:</p> <ul style="list-style-type: none"> <li>• Ability of students to script code chunks with a view to automate PE computational task.</li> <li>• <b>Example:</b> A Python script to request reservoir properties and compute reservoir volumetrics such as BV, PV, HCPV and STOIPP.</li> </ul>
LO7: Students should be able to develop algorithms and Python scripts to execute integrated petroleum engineering computational workflows.	<p><b>Acceptable evidence</b> of this learning objective will be:</p> <ul style="list-style-type: none"> <li>• Ability of students to interpret a given PE problem statement, formulate algorithms and write a wholistic Python script to solve the problem.</li> <li>• <b>Example:</b> A script that completely implements the oil material balance equation for a discretized reservoir. The inputs being gridblock rock and fluid properties as well as gridblock pressure values per time. The output being various reservoir performance parameters such as well flowrates, cumulative oil produced, average reservoir pressure, etc.</li> </ul>

These assessment rubrics, as presented in Table 1, are deployed in both formative assessments (take-home programming assignments, in-class quizzes and tests) and summative assessments (final exams). The programming assignments were given in pursuit of *Active Learning* which is known to enhance deeper learning outcomes (Enomoto, Warner and Nigaard, 2022). In order to situate the examination in a quasi-real-world context, the candidates are asked to assume roles such as Reservoir Engineering Intern working with a Senior Reservoir Engineering Advisor. A sample programming assignment can be viewed [here](#) while its expected solution (Python script) can be viewed [here](#). Also, sample examination questions can be accessed [here](#); with the solutions available [here](#).

### Pedagogy: Instructional Strategies

Finally in this course module design, we now present various efforts and initiatives, undertaken by the team of instructors and supported by the University, to guide learners through the process of attaining the desired outcomes. Specifically, we present the lesson sequence, the learning activities, the learning resources, learning environment and other strategies adopted in implementing the course. These efforts are suited to achieve the desired learning outcomes.



**The Preambles.** Prior to the core lessons, the course starts with a couple of activities as preambles. First, the delivery of the module opens with an exciting presentation on the skills for emerging oil/gas opportunities in data analytics (DA), machine learning (ML) and artificial intelligence (AI). While this course is not about DA, ML and AI, the Python coding skills taught in this course is required of anyone aspiring to take advantage of those emerging opportunities. Hence, the presentation serves as an appetizer to stimulate the students' interest in the course. Figure 2 is a snapshot of the Outline slide of the presentation as a preview of the contents. The full presentation can be accessed [here](#). In concluding the preambles, a guide to the installation of all the toolboxes needed for the course is presented. Students are guided through the installation of Python 3, the primary tool for this course. Additionally, instructions on the installation of GitHub Desktop as well as creation of GitHub account are also presented. GitHub is the online platform used for collaboration and version control (discussed below) in this course. As a graded assignment, each student is made to install all these tools on their personal computers.



Figure 2—Outline of the 'Appetizer' Presentation

**The Lesson Sequence.** The lesson sequence starts with a gentle introduction to computer programming is presented. This becomes important as the course assumes students have no prior programming experience. The introduction draws a striking similarity between a programming language and a natural language. Also, some common fundamental patterns in a typical program are presented. At its core, the module teaches the basic features of the Python programming language – input/output statements, objects (values, variables, keywords and statements). Various control structures such as conditional execution (if...), alternative execution (if...else), definite repetitive loops (for...) and indefinite repetitive loops (while...) are presented. The concept of code re-usability encapsulated as custom functions is also presented. The final units of the course module are dedicated to the various Python's in-built multi-valued data structures (lists, tuples and dictionaries).

**Conceptual Applications to Petroleum Engineering.** The entire lesson units are enriched with simple conceptual petroleum engineering applications and examples. It is with such examples that various coding concept are presented. Examples and applications have been selected from fluid property correlations, reservoir oil volumetrics, material balance equation (MBE) simulations, reservoir characterization and reservoir flow simulation. Some specific cases are here mentioned. Reservoir rock and fluid properties have been used as examples while presenting variable creation and value assignment. Choosing between the

two regimes of the variation of oil formation volume factor ( $B_o$ ) with pressure have been used to illustrate the if...else control structure. Looping through the gridblocks of a discretized reservoir model has been demonstrated as a for... loop. The reservoir simulation task of updating  $B_o$  values at series of decreasing reservoir pressure until bubble point attainment is presented as a case of the while... loop. The use of Python lists has been exemplified with the storage of reservoir simulation grid properties. Furthermore, Python dictionaries have been presented as a way to tag simulation outputs with respective gridblock identifiers. Over the two sessions of its implementation, the course has built a library of custom functions (codenamed peteng) scripted in the class to perform some simple PE computations. A snapshot of a section of the library is presented here as Figure 3 while the full content of the library is accessible [here](#). It is expected that more custom functions would be added to this library in the coming sessions of offering this course. It is noteworthy to state here that numerous application examples are available in an on-going project aiming at a pedagogical unboxing of reservoir simulation with Python (Mosobalaje and Olayemi, 2023).

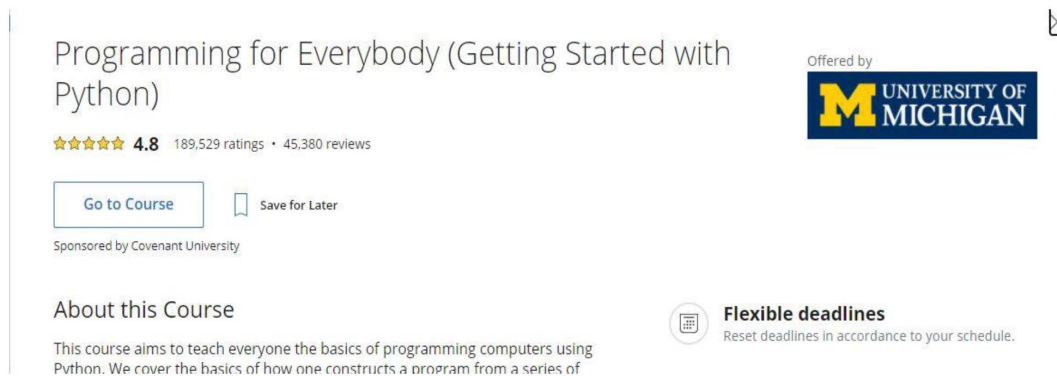
```

1  # ... to the only wise God
2
3  # This is the home of functions that implements simple petroleum engineering computations.
4
5  ##### A function to compute real gas density #####
6  # Note: pressure must be in psia and temperature in degree Rankine
7
8  def gas_density(gravity, pressure = 14.7, temperature = 520, z = 1):
9      density = (2.70*pressure*gravity)/(z*temperature)
10     return round(density, 4)
11
12 # round(gas_density(0.78), 3)
13
14 ##### A function to estimate bubble point pressure, pb #####
15 # Note: this function only works if solution gas-oil ratio at a pressure above bubble point (i.e. Rsi (=Rsb)) is known
16 # Note that temperature is in degree Fahrenheit
17
18 def bubble_pressure(temperature, pressure, gas_gravity, oil_gravity, rsb):
19     api = (141.5/oil_gravity)-131.5
20     y = (0.00091*temperature)-(0.0125*api)
21     pb = (18*(10**y))*((rsb/gas_gravity)**0.83)
22     return round(pb,2)

```

Figure 3—A snapshot of the peteng library

**Blended Learning Approach.** In the wake of the outbreak of COVID-19, Covenant University adopted the blended learning approach that combines the conventional classroom-based learning with online educational opportunities. Specifically, the University subscribed to Coursera – the US-based massive open online course (MOOC) provider. With the subscription, the University encourages instructors to identify relevant Coursera courses to blend with their in-house courses. In response to this initiative, the team of instructors for this course adopted the most popular Python MOOC: Programming for Everybody (Getting Started with Python) for blending with this course. This MOOC is offered on Coursera by the University of Michigan. A snapshot of the Coursera page featuring this MOOC is presented here as Figure 4. In order to enforce the blending, the certificate of completion of the MOOC is listed as part of the compulsory assessment for all registered students of this course. Over the past two offerings of the course, about a hundred students have been guided to obtain the MOOC certificate – this enhances their employability, particularly as they go in search of internship positions.




Programming for Everybody (Getting Started with Python)

★★★★★ 4.8 189,529 ratings • 45,380 reviews

[Go to Course](#) [Save for Later](#)

Sponsored by Covenant University

Offered by  UNIVERSITY OF MICHIGAN

About this Course

This course aims to teach everyone the basics of programming computers using Python. We cover the basics of how one constructs a program from a series of


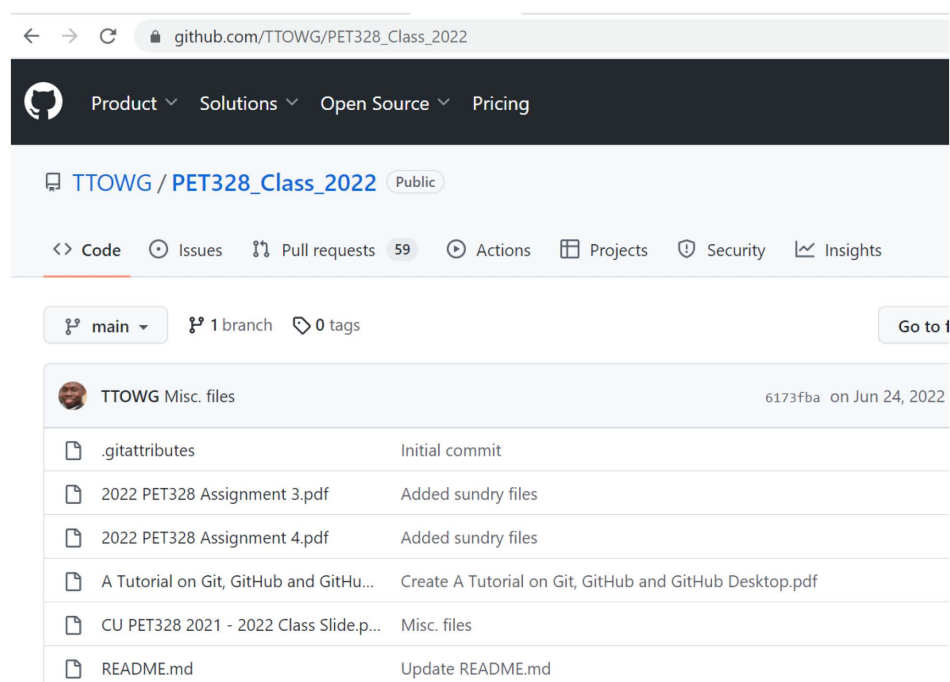
 **Flexible deadlines**  
Reset deadlines in accordance to your schedule.

Figure 4—A snapshot of the Coursera page featuring the blended MOOC

**The Use of GitHub for Collaboration.** GitHub is an online platform through which project files (especially code scripts) are hosted on remote servers and made accessible to project team members, for collaboration purposes. In order to simulate real-life scenario, this course assumes that all participants (students and instructors) belong to a team and need to collaborate on project files. First, in order to drive the idea, this course features a tutorial training on the use of GitHub and GitHub Desktop tools. The tutorial is accessible [here](#). For each session that the course has been offered, a GitHub repository is created to host all course materials: slides, demo scripts, programming assignment scripts, peteng library. The repository for the 2020/2021 session is accessible [here](#) while that of the 2021/2022 session is available [here](#). A snapshot of the 2021/2022 repository is here presented as Figure 5. Every student is made to create a GitHub account, to fork and clone the repository. With this, the instructors work on demo or assignment files, commit and push changes to the main branch of the repository. Students in turn updates their copies of the repository by accepting pull requests originating from the instructor's main branch. Specifically, for programming assignments, students edit a pre-configured code script, commit and push the edits to their branch of the repository, and ultimately submit the URL to their branch.



github.com/TTOWG/PET328\_Class\_2022

Product Solutions Open Source Pricing

TTOWG / PET328\_Class\_2022 Public

<> Code Issues Pull requests 59 Actions Projects Security Insights

main 1 branch 0 tags Go to 1

TTOWG Misc. files 6173fba on Jun 24, 2022

.gitattributes	Initial commit
2022 PET328 Assignment 3.pdf	Added sundry files
2022 PET328 Assignment 4.pdf	Added sundry files
A Tutorial on Git, GitHub and GitHu...	Create A Tutorial on Git, GitHub and GitHub Desktop.pdf
CU PET328 2021 - 2022 Class Slide.p...	Misc. files
README.md	Update README.md

Figure 5—A snapshot of the 2021/2022 course repository on GitHub



**Open Access to Learning Resources.** Covenant University fully subscribed to the open access (OA) policy both in regards to research publications and pedagogy materials (Omonhinmin et al., 2014a, 2014b). In line with this institutional policy, all materials related to this course module are all freely accessible on the GitHub repositories created for this course, [here](#) and [here](#); for the 2021 and 2022 offerings, respectively. The repositories are licensed under the GNU General Public Licence v3.0. The free access to these materials is an accomplishment of a key purpose of this publication – to provide a guide for the adoption of this course initiatives across petroleum engineering programs in Nigeria. Materials deposited in the repositories include: lecture slides, demo scripts, the peteng library, custom function files, PE workflow algorithms, programming assignments, sample tests and exam questions and solutions, tutorial slides on the use GitHub etc. Additionally, for easy navigation, hyperlinks to these resources are included in the lecture notes. With this, a student gets redirected to GitHub repository right from the PDF notes upon clicking the hyperlinks.

## Future Prospects

This course is only the first in a series of such initiatives. In addition to this stand-alone introductory course module, the department hopes to incorporate Python coding into many of its course modules, particularly as computational tools. Also, a future course module on petroleum data analytics and machine learning (PDA&ML) is being proposed. The proposal is part of the 30% curricula being allowed to be independently added at individual universities in the Nigerian Universities Commission's Core Curriculum and Minimum Academic Standards, CCMAS (Nigerian Universities Commission, 2022). A post-graduate course module in this regard is also being considered in the nearest future. Ultimately, we hope to improve on this initiative and offer it as a short-course for industry professionals.

In future publications regarding this initiative, we hope to present data on students' performance in this course. We also look forward to measuring the impacts of this course on students' final-year research projects, follow-up courses, employability, internship placements, and career progression.

## Conclusion

A recent initiative at incorporating Python coding into the petroleum engineering program in Covenant University has been comprehensively documented in this paper. The motivation for this initiative is succinctly highlighted herein. A comprehensive layout of the curriculum design, in terms of content, assessment and pedagogy has been presented. In presenting the layout, the UbD Framework is utilized. Open access to the course materials has been provided through hyperlinks embedded in this paper. Finally, the future outlook of this initiative has been set out. While we solicit feedbacks from peers, we recommend the adoption of this initiative to other petroleum engineering programs in Nigeria and beyond.

## Acknowledgement

The authors wish to appreciate the management of Covenant University for the permission to publish this paper. Also, the first author appreciates Dr. Moses Olayemi for painstaking tutorials on curriculum design concepts. We also acknowledge the cooperation and commitment of all our students who participated in the course module.

## References

- Anderson, L. W. and Krathwohl, D. R. 2001. A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives. Longman
- Du, D., Zhang, X., Guo, Q., Zhang, B., Zhang, G. 2020. Smart Oilfield Technology. In: Lin, J. (eds) *Proceedings of the International Field Exploration and Development Conference*
- Elochukwu, O.H., Joukarborazjany, M. and Attarhamed, F. 2013, Incorporating Oil and Gas Softwares into Petroleum Engineering Education Curriculum, *Engineering Science and Technology*, Vol. 3 Number 1, February, 2013.

- Enomoto, K., Warner, R. and Nygaard, C. 2022. Active Learning in Higher Education: Student Engagement and Deeper Learning Outcomes. Libri Publishing.
- Feder, Judy. 2019. As Industry Changes, So Does Petroleum Engineering Education. *J Pet Technol* **71** (2019): 44–48. doi: <https://doi.org/10.2118/1219-0044-JPT>
- G. Wiggins and J. McTighe. 2005. Understanding by design. ASCD.
- IBM. 2020. How AI can Pump New Life into Oilfields. *IBM Institute for Business Value*. Available at <https://www.ibm.com/downloads/cas/5BNKGNLE#:~:text=Leveraging%20AI%20can%20enable%20automatic,develop%20estimation%20and%20prediction%20models>.
- Mathieson, Derek, Meehan, D. Nathan, and Jeff Potts. 2019. The End of Petroleum Engineering as We Know It. Presented at the SPE Middle East Oil and Gas Show and Conference, Manama, Bahrain, March 2019. doi: <https://doi.org/10.2118/194746-MS>
- Mosobalaje, O. O. and Olayemi, M. 2023. A Pedagogical Unboxing of Reservoir Simulation with Python – Backward Design of Course Content, Assessment, and Pedagogy (CAP). Presented at the 2023 American Society of Engineering Education (ASEE) Annual Conference and Exposition. Available at: [https://nemo.asee.org/file\\_server/papers/attachment/file/0015/1175/ASEE\\_2023\\_Mosobalaje\\_Unboxing.pdf](https://nemo.asee.org/file_server/papers/attachment/file/0015/1175/ASEE_2023_Mosobalaje_Unboxing.pdf)
- National Universities Commission. 2022. Engineering and Technology, *NUCCMAS*. Available at: <https://nuc-ccmas.ng/engineering-and-technology/> (accessed March. 25, 2023).
- Omonhinmin, C.A., Agbaike, E. and Atayero, A.A. 2014b. Implementing Open Access in a Private Nigerian University: A case study of Covenant University. Presented at The International Conference on Web and Open Access to Learning. Available at: <https://doi.org/10.1109/ICWOAL.2014.7009198>.
- Omonhinmin, C.A., Omotosho, O.E., Akomolafe, A. and Atayero, A.A. 2014a. Policy for development and use of open educational resources in covenant university: an open access policy in Covenant University. Presented at The International Conference on Web and Open Access to Learning. Available at: <https://doi.org/10.1109/ICWOAL.2014.7009198>.
- PYPL. 2023. Popularity of Programming Language (PYPL) Index. Available at <https://pypl.github.io/PYPL.html>. Accessed: March, 22, 2023.
- Python Software Foundation. 2023. About Python. Available at <https://www.python.org/about/> Accessed: March, 22, 2023.
- Qadir, J., & Shafi, A., & Al-Fuqaha, A., & Taha, A. M., & Yau, K. A., & Ponciano, J., & Hussain, S., & Imran, M. A., & Sheikh Muhammad, S., & Rais, R. N. B., & Rashid, M., & Tan, B. L. 2020. Outcome-based (Engineering) Education (OBE): International Accreditation Practices Paper presented at 2020 ASEE Virtual Annual Conference Content Access, Virtual Online. [10.18260/1-2—35020](https://doi.org/10.18260/1-2-35020).
- Society of Petroleum Engineers (SPE). 2021. Competency Matrix for Data Science and Engineering Analytics. Available at: <https://www.spe.org/en/training/dsea-competency-matrices/> (accessed May. 31, 2023).