Cloud Computing Engineering
e516

Gregor von Laszewski

Editor

laszewski@gmail.com

https://github.com/cyberaide/bookmanager

August 25, 2019 - 06:58 AM

Created by Cloudmesh & Cyberaide Bookmanager, https://github.com/cyberaide/bookmanager

Cloud Computing Engineering

Gregor von Laszewski

Cloud Computing Engineering

1 PREFACE

Sun Aug 25 06:58:49 EDT 2019

1.1 DISCLAIMER

This book has been generated with Cyberaide Bookmanager.

Bookmanager is a tool to create a publication from a number of sources on the internet. It is especially useful to create customized books, lecture notes, or handouts. Content is best integrated in markdown format as it is very fast to produce the output.

Bookmanager has been developed based on our experience over the last 3 years with a more sophisticated approach. Bookmanager takes the lessons from this approach and distributes a tool that can easily be used by others.

The following shields provide some information about it. Feel free to click on them.

1.1.1 Acknowledgment

If you use bookmanager to produce a document you must include the following acknowledgement.

> *"This document was produced with Cyberaide Bookmanager developed by Gregor von Laszewski available at https://pypi.python.org/pypi/cyberaide-bookmanager. It is in the responsibility of the user to make sure an author acknowledgement section is included in your document. Copyright verification of content included in a book is responsibility of the book editor."*

The bibtex entry is

```
@Misc{www-cyberaide-bookmanager,
  author =   {Gregor von Laszewski},
```

```
  title =    {{Cyberaide Book Manager}},
  howpublished = {pypi},
  month =    apr,
  year =     2019,
  url={https://pypi.org/project/cyberaide-bookmanager/}
}
```

1.1.2 Extensions

We are happy to discuss with you bugs, issues and ideas for enhancements. Please use the convenient github issues at

- https://github.com/cyberaide/bookmanager/issues

Please do not file with us issues that relate to an editors book. They will provide you with their own mechanism on how to correct their content.

2 SYLLABUS

2.1 SYLLABUS ENGINEERING CLOUD COMPUTING

Learning Objectives

- Get a quick overview what the class is about

2.1.1 Instructor

- Gregor von Laszewski laszewski@gmail.com
- Office hours: By appointment

2.1.2 Audience

- We recommend you know one programming language.
- Knowledge of python is of advantage but not required. Python is easy.

2.1.3 Course summary

This class will introduce you to state-of-the-art cloud computing concepts an engineering approaches. This will include virtual machines, containers and Map/Reduce. The course will have a Lab in which you can practically explore these concepts. You will for example have the option to create a cloud as part of this cours and explore cloud computing tools and frameworks.

2.1.4 References

The course does not have required readings. We will provide the following references as pointers to what we will discuss:

2.1.5 Tools

You will be required to have a computer to log into the cloud. We will give you access to an OpenStack cloud. Access to Azure, AWS, Google and others can be achieved through their free tier,

2.1.6 Course Schedule

Week	References
1	Course Introduction
2	Cloud Data Centers
3	Python for Cloud Computing, Start of Project Selection
4	Cloud Architectures
5	Virtualization I - OpenStack
6	Virtualization II - AWS, Azure, Google
7	Multi Cloud Environment
8	Cloud Technology Presentation - Project Review
9	Containers - Docker, Kubernetes
10	Map Reduce
11	Messaging
12	REST
13	GO
14	Project Work
15	Projects Due
16	Project Improvements

For each of the topics you will find one or more relevant chapters or sections in our online book.

2.1.7 Attendance

Attendance accounts for 10% of your final grade. If you need to skip class for any reason, you need to notify the instructor and TAs

2.1.8 Assignments

This course will not have exams. Instead, we have the following graded assignment categories:

- Lab Assignments (pass/fail) are assignments that will be conducted on a weekly basis. They will help you making sure you not only understand the material theoretically, but try them out.

- Cloud Technology Review and Examples (Graded): (This can be substituted for more programming in your project). This is a document about a Cloud technology that is not yet included in our handbook to introduce an interested party to it. It should not contain advertisements but be a rational description of the technology with examples that you have tried yourself. You will have to give a non plagiarized presentation and document about it.

- Project Assignments (Graded): The most important part of the class for which you will be working throughout the semester. Up to three students can work in a project. In case of group projects, the project deliverables are increased.

The project has three submissions that are spread throughout the semester. Each submission builds on the previous one and modifies previous documents into a consistent project report and documentation for your project.

- Project Outline

  A description of what your project is about and how it relates to cloud computing and address:

  - What is the problem you try to address?
  - What are you doing to address this problem?
  - How are you addressing this problem?
  - What is the architecture that adresses the problem that you will implement?

- Code and Documentation Review

  - You will be asked to have a meeting with the TA’s and/or instructor to showcase your code and have at least one review prior to your final

submission.

    - This will take usually take place through the Lab hours on regular basis.

    - A first project discussion must have been done at least once at midterm time.

- Final Project Submission

    - All code and documentation must be checked into GitHub well before the semester is over. This allows us to give you feedback for improvements.

Please note that the syllabus is subject to change. Students in this class often come from a wide variety of backgrounds and experiences. As such, the instructor reserves the right to change the content of the course to accommodate the needs and expectations of the students.

2.1.9 Project Examples

- https://cloudmesh.github.io/cloudmesh-manual/projects/index.html

2.1.10 Statement on Academic Misconduct

Students will be expected to uphold and maintain academic and professional honesty and integrity as outlined in the Code of Student Rights, Responsibilities, and Conduct. Cases of academic misconduct will be handled according to the student disciplinary procedures described in IU's policies.

3 OVERVIEW

3.1 COMMUNICATION

3.1.1 Piazza

This class uses piazza for communication. It is your responsibility to enroll in the Piazza for the class. A link to our PIazza is provided in CANVAS. PIazza works just like a forum in which you can ask questions or post notes. In case of a question STudent sand teachers can formulate answers. A thread system in the question can be used to gather the answer.

- https://piazza.com/iu/fall2019/e516fall19

You will get a grade for participation in the class discussions on Piazza. Please note that we do not recommend you go from CANVAS to piazza as CANVAS has a bad layout and in fact does not allow you to see the Piazza Hyperlinks. Instead go to Piazza directly.

3.1.2 Class Resources

All class material will be posted at

- https://piazza.com/iu/fall2019/e516fall19/resources

3.1.3 Online Meeting

If you are an online student:

A poll has been posted that you must fill out before Friday of the first week of the semester for possible meeting times.

- https://piazza.com/class/jzkfveoqwri3e4?cid=8

The link to the meeting information is at

- https://piazza.com/class/jzkfveoqwri3e4?cid=7

3.1.4 Assignments

All Assignments are links in the piazza resource page as a simple link to CANVAS in the resources page:

- https://piazza.com/iu/fall2019/e516fall19/resources

3.1.5 Post Your Bio

To assure that you have access to Piazza and can post to it, please, post a formal Bio.

3.1.6 Evolving Document

This document is improved throughout the semester with weekly activities. We recommend that you regular download a new version. Be aware that some browsers or ePub readers may cache the previous version. Thus make sure to check out the newest version.

3.1.7 Books

This class does not use books from publishers as cloud computing is constantly changing and by the time you place the purchase order, the book may already be outdated. Instead we have prepared online books that are constantly evolving.

We like to illustrate on a simple example on what happened to us in the past. We prepared with great detail information about Azure. However throughout the semester the way on how to interact with it was changed dramatically. Now if you would have had a book, the entire chapter about Azure would have been outdated. However, within less than a week, we were able to replace the content so everyone in class can benefit from our updates very quickly. Not only that, because our materil is online you will likely fiond an updated versiso in future even after your class is completed. No printed text book can provide this service.

We have the following books in ePub and PDF available. The ePub version is our prefered version:

- List of Books

The weekly Lecture notes are contained at

- e516: Cloud Computing Engineering

This page will bring you to the following books that we use as part of e516. It will also clearly specify which assignments you need to do as part of the weekly Lab activities.

- Cloud Computing
- Linux
- Python
- Markdown

3.1.8 You Get Credit for Improving the Books

Please note that improving the document in GitHub via pull requests will get you credits. This is an easy opportunity for you to get an excellent grade.

3.1.9 Ongoing Agenda

⚠ We will add here on a weely basis the topics and Labs that we will cover in the commig week(s). Please be awere that jsut as in regular lectures we add the agenda items and follow the Syllabus. We may change the order of the topics covered in the syllabus as appropriate.

3.2 Quick Tips

3.2.1 Requirements

We recommend that you know one programming language. Although most activities are done in Python, this programming language does not have to be Python as it can easily be learned throughout the semester. No background in cloud computing is needed.

3.2.2 Time Commitment

Any class at an university requires a significant time commitment. Due to the different background the students have it is difficult to predict the actual time needed. On average we see that students spend 6 hours on the class if they do participate on a weekly basis. Students with little programming experience spend up to 12 hours.

3.2.3 Course Material List

Course material will be distributed as ePubs, PDF, Video or Presentations. However you will be required to research some topics on the on the internet as Cloud Computing ia arapidly evolving field and we find the most up to date information on the Web. We also ask you to help us updating the ePub and to use additional resources as appropriate.

The list includes:

- Cloud Computing, Gregor von Laszewski, Ed. 2019 [1]
- Cloud Technologies, Gregor von Laszewski [2]
- Project Report Format, Gregor von Laszewski [3]
- Introduction to Python [4]
- Linux for Cloud Computing [5]
- Scientific Writing with Markdown [6]

3.2.4 Help

If you take our class, please use piazza to ask for help. This is important as questions may be answered by different Teaching Assistents (TAs) based on expertise. Please, do not send e-mail to the instructors. TAs are not allowed to answer e-mail send to them personally.

3.2.5 How to Take this Class

This class is attended by students with greatly different backgrounds and time schedules. To be most flexible and adress all students there are two different ways on how you can take this class.

- Way 1: *Free form.* Here you simply look at the Syllabus table for the semester and identify whatever section you feel like reading (when it becomes available). However, make sure you conduct our weekly **Lab activities**.

- Way 2: *Chronological order*. The lecture notes are ordered
- chronological. Thus you can follow our lecture alos in chronological

- order.

Please note that we have set aside a recommended set of weekly Lab activities. The activities are pass-fail and will be integrated in your grade. You are certainly allowed to work ahead, but please be aware that based on feedback and observation we may make modifications to the Labs.

Typically, Lab activities are supposed to be completed within one week as it alerts us of problems you might have that we can than address. This assures us that we know you will have no issues with your project.

Lab activities will not receive any credit if you are a residential student and the activity has not been completed within one week. However, residential students will get two **Delay a Lab for One Week** passes that you can apply to any of the Labs and still get credit.

If you are an online student we recommend that you finish the Labs also in one week. However, you will get eight **Delay a Lab for One Week** passes that you can apply to any of the Labs.

Please note that if you would need to postone a lab for two weeks, you need to use two passes. Lab passes expire one month before the last day of class. You will have to complete all labs by that time. No credit will be given at that time if this deadline is missed for any delayed Labs as TAs must focus their attention on project support.

Lab passes do not apply to other assignments and due dates.

3.2.6 Assignments

Besides the Lab's, we have only two main assignments in this class. The Lab's will prepare you towards achieving these assignments.

3.2.6.1 Technology Review

> *Students doing projects related to cloudmesh are exempt from writing a technology review, but are expected to do more programming and making sure the project has a manual and proper documentation.*

As part of cloud engineering you will be exposed to a large set of technologies. To sharpen your skills in analyzing and evaluating these technologies, you will be asked to prepare a technology review that is being added to a class proceedings.

This includes a substantial non-plagiarized document that can be added as a chapter to the lecture notes. The review must be done on a topic that is not yet included in our book. The review will not include advertisement statements form those that have developed the technology, but will qualitatively describe the technology and potentially contrast it to other related technologies. In addition you will have to develop an example showcasing how to use the technology. The minimal length of a review is about 800 words.

An example for such a section is

- GraphQL in the Cloud Computing book

Alternatively you can prepare several different smaller sections (at least 5) that may not have an example in it but are more of descriptive nature. Sample sections contributed by students include:

- Section Microsoft Nafik Data Center in the Datacenter Chapter
- Section Lambda Expressions in *Introduction to Python*

> ⚠ *It is expected from you that you self identify a section yourself, as this shows competence in the area of cloud computing. If however you do not know what to select, you must attend an online hour with us in which we identify a topic with you. Technologies that are not*

> *repeatable due to enormous cost or licensing issues need to get prior approval.*

3.2.6.2 Project

The objective of the project is to define a clear problem statement and create a framework to address that problem as it relates to cloud computing.

A project is the major activity that you chose as part of your class. This includes a project report* or *manual* and working project code. You will create a significant non-trivial project related to cloud computing and cloud engineering. Up to three students can collaborate. The project could be built on top of a previous project but must have significant additions or modifications. If a previous project is used, a detailed discussion is to be held on what has been improved and is different.

In this class it is especially important to address the reproducibility of the deployment. A test and benchmark, possibly including a *downloadable* dataset, must be used to verify the correctness of your approach.

3.2.6.2.1 License

All projects are developed under an open source license, such as the Apache 2.0 License. You will be required to add a LICENCE.txt file and describe how other software, if used, can be reused in your project. If your project uses different licenses, please add a README.md file that describes which packages are used and what licenses these packages have.

3.2.6.2.2 Project Report

A project report is to be delivered and continuously improved throughout the semester in GitHub. It includes not just the analysis of a topic, but a short description of the Architecture and code, with **benchmarks** and demonstrated use. Obviously it is longer than a term paper and includes descriptions about reproducibility of the application. A README.md is provided that describes how others can reproduce your project and run it. Remember that tables and figures do not count towards the paper length. The following minimal length is

required:

- 800 words, one student in the project
- 1200 words, two students in the project
- 1400 words, three students in the project

Projects with more students are expected to do more programming. The report is written in markdown and checked into GitHub. The report will be made available in a class proceedings. A Report could be substituted by a manual and benchmarks. In this case a one page extended abstract has to be written, that includes the ling to the manual.

For certain projects, the requirement of a report can be waved or is significantly reduced while replacing it with more programming activities. This includes

- Any project that enhances cloudmesh
- Building a large cloud cluster with Raspberry Pi's
- Any Application project showcasing NIST big data reference architecture use (there is a hard deadline of the NIST project by Dec 1st).

However you still have to do a manual and usage examples, benchmarks and `pytest`s for them.

3.2.6.2.3 Project Code

You are expected to deliver a **documented** and **reproducible** code with unit tests that allows a TA to replicate the project with ease. In case you use vm or container images, they must be created from **scratch locally** and may not be uploaded to services such as DockerHub. You can, however, reuse approved vendor uploaded images such as from ubuntu or centos or other linux distributions. All code, scripts, and documentation must be uploaded to github.com under the class specific GitHub directory.

3.2.6.2.4 Project Data

Data is to be hosted on IU's Google drive, if needed. If you have larger data, it should be downloaded from the internet. It is your responsibility to develop a download program. The data **must** not be stored in GitHub. You are expected to

write a python program that downloads the data either from the Web or IU's data storage.

3.2.6.2.5 Work Breakdown

This is an appendix to the document that describes in a bullet list who did what in the project. If you are a team of one such a section is not needed. This section comes after the references. It does not count towards the page length of the document. It must includes explicit URLs to the git history that documents the statistics to demonstrate more than one student has worked on the project. If you can not provide such a statistic or all check-ins have been made by a single student, the project has shown that they have not properly used git. Thus, points will be deducted from the project. Furthermore, if we detect that a student has not contributed to a project, we may invite the student to give a detailed oral presentation of the project including a demonstration of the examples in real time.

3.2.6.2.6 Bibliography

All bibliography has to be provided in a BibTex file that **must** either be validated with **jabref** or with **emacs**. Please be advised doing references correctly takes some time so you want to do this early and throughout the semester. What would take less than 5 minutes a week, could quickly add up to multiple hours at the end of the semester. Please note that exports of Endnote or other bibliography management tools do not lead to properly formatted bibtex files, despite their claims of doing so. You will have to clean them up and we recommend to do it the other way around. Hence, the easiest way to manage your bibliography is with *jabref* or *emacs*. Make sure **labels** only include characters from [a-zA-Z0-9-]. Use dashes and not underscore and colons (`_`,`:`) in the label. Your labels must be meaningful and unique. We will deduct points if you submit an invalid *BibTex* file to GitHub. So please make sure your file is validated. You can even create your own checks with tools such as `biber`.

We will teach you what to do it is easy.

3.2.6.2.7 Reproducibility

In general, any project must be deployable by the TA. If it takes hours to deploy

your project, please talk to us before final submission. This should not be the case. Also, if it takes 100 steps, we are sure you can automate them … as you are likely doing something wrong or have not thought about cloud computing where we tend to automate most of the steps.

You have plenty of time to execute a wonderful project but you need to work consistently on it. Starting one week before the deadline will not work.

The bets way to asure reproducibility is to use `pytest`. We will discuss how to do that in class.

We will teach you what to do it is easy.

3.2.6.2.8 List of Deliverables

In general your deliverables will include the following (We will address and explain them in a Lab):

- Provide benchmarks.
- Take results in a cloud services and your local PC (ex: Chameleon Cloud, echo kubernetes). Make sure your system can be created and deployed based on your documentation.
- Each team member must provide a benchmark on their computer and a cloud IaaS, where the cloud is different from each team member.
- We require you to write one or more pytest's that deploys, run, kill, view, clean that deploys your environment, runs application, kills it, views the result and cleans up after wards.
- For python use a requirements.txt file and develop a `setup.py` so your code can be installed with `pip install .`
- For docker use a Dockerfile

We will teach you what to do …

3.2.6.2.9 Example Outline of a Report

- (If not exempt) write a report that typically includes the following sections:
    - Abstract

    - Introduction
    - Design
        - Architecture
    - Implementation
        - Technologies Used
    - Results
        - Deployment Benchmarks
        - Application Benchmarks
    - (Limitations)
    - Conclusion
    - (Work Breakdown)

- Your report will **not** have a *Future Work* section as this implies that you will do work in future and your paper is incomplete. Hence we would not grade it. Instead, you can use an optional “Limitations” section.

- Do communicate your status and add a *Workbreakdown* section in which you outline which tasks need to be done and by whom in case of a group project. Once you have done a task simply include maker a task as follows

  ```
  * [done, Gregor] This was gregors task to showcase how to mark it
  ```

In case you have an exemption for the project report you need to use `sphinx` and document your code as part of a manual. We will explain the details in one of our labs.

3.2.7 Submission

All submissions are conducted via GitHub if not otherwise instructed. Technology reviews are to be added to the `book` GitHub repo with the help of pull requests. The TA’s will work with you to integrate them.

As we are working continuously throughout the semester you must indicate your activities in a README.yml file in your GitHub repo. The GitHub Repo we will define for you in the first 2 weeks of the semester.

An example for the README.yaml file is shown next

```
---
owner:
```

```
  firstname: Gregor
  lastname: von Laszewski
  hid: fa18-523-00
  community: i523
  semester: fa18
chapter:
  - keyword: IoT
    title: Role of Big Data in IoT
    url: https://github.com/cloudmesh-community/fa18-523-00/blob/master/chapter1/paper.md
    group: fa18-523-00 fa18-523-01
  - keyword: Datacenter
    title: Green IT data centeres in the US
    url: https://github.com/cloudmesh-community/fa18-523-00/blob/master/chapter2/paper.md
    group: fa18-523-00
project:
  - keyword: Cloud Cluster
    title: Raspberry PI Cloud Cluster
    url: https://github.com/cloudmesh-community/fa18-523-00/tree/master/project-report
    group: fa18-523-00 fa18-523-01
    code: TBD
```

You **MUST** run `yamllint` on the `README.yml` file. YAML errors will cause point deductions. Any invalid yaml file will result in point deductions. Please keep your yaml file valid at any time. Our scripts depend on it. The yaml file will also be used to create a list for TAs to review your deliverables. If it's not in the yaml file it will not be reviewed. Please note that it is not sufficient to just run `yamllint`, but to compare your yaml file carefully with the `README.yml` examples. Make sure you do the indentation with 2 spaces, do not use the TAB character and make sure you use the list and attribute organization with proper dash placement. Work with the TAs if you have difficulties. If you copy, only copy from the raw content in GitHub. If you work on more

We will teach you what to do it is easy.

3.2.8 Bonus Projects

This class will not have any bonus projects, as all additional activities should be put in your project or chapter/review contribution. However, we will recognize extraordinary efforts in these activities.

3.2.9 Participation

In addition to these artifacts, there will also be a participation component in class that will be determined based on your productive contributions to piazza to help others that have questions and contributions to the books to, for example, improve sections with spelling, grammer or content. We can see from the GitHub history if you conducted such improvements. Make sure that technical contributions, work on all OSes and are not just targeting a single OS if the

improvement is of general nature (exceptions apply).

4 WEEKLY AGENDA

4.1 WEEK 1: COURSE INTRODUCTION

4.1.1 Assignments

All graded class assignments are posted at

- https://iu.instructure.com/courses/1822529/assignments

We provide in this document all activities we do in each week

4.1.2 Lab Activities

- Post your professional bio to piazza
- Setup your computer
- Create the cloud accounds and fill out the form
- Install python on your computer (python 3.7.3 or 3.7.4)
- Update pip

4.1.2.1 Account Creation

As part of the class you will need a number of accounts. This includes:

Required accounts:

- piazza.com (used for communication)
- github.com (used for project and other class artifacts)
- chameleoncloud.org (free cloud account)

After you created the accounts, please fill out the following form so we can set up the class accounts and in case of github we create you a class repository for you.

- Cloud Accounts

Optional accounts (apply once you need them, some are time limited):

- google.com (optional)
- aws.com (optional)
- azure.com (optional)
- Watson from IBM (optional)
- google Iaas (optional)

A survey is to be filled out in the first week of class. It includes your github.com account that we need to create your github directory in which you will submit your open source project.

4.1.2.2 Bio

This activity serves two purposes. First, it tells us we can communicate with you within piazza, and second, you can introduce yourself to others in piazza to potentially build project or study teams.

4.1.2.3 Python 3.7.4

Please set up a computer on which you can do Python development. We recommend that you use python from http://python.org. We will not provide any support for conda, as conda has hundrest of libraries that we are not interested in using. Also note that conda modifies your environment without telling you. Certainly you can use virtualbox, or containers in case you like to isolate your development environment from your other systems. Please remember that conda has significant disadvantages of often working with outdated libraries. As developer of future software you may want to avoid this. If you use python from python.org we recommend that you use venv.

4.1.2.4 Questionnaire

In this activity you will be filling out a form with information about you so we can asses show to best integrate you in this class. It is not important that you know any of the technologies we ask you about.

- Background Questionnaire

5 APPENDIX

5.1 ePub Readers

This document is distributed in ePub format. Every OS has a suitable ePub reader to view the document. Such readers can also be integrated into a Web browser so that when you click on an ePub it is automatically openedin your browser. As we use ePubs the document can be scaled based on the users prefernc If you ever see a content that does not fit on a page we recommend you zoom out to make sure you can see the entire content.

We have made good experiences with the following readers:

- **macOSX**: Books, which is a build in ebook reader
- **Windows 10**: Microsoft edge, but it must be the newest version, as older versions have bugs. Alternatively use calibre
- **Linux**: calibre

If your have an iPad or Tablet with enough memory, you may also be able to use them.

5.2 Corrections

The material collected in this document is managed in

- https://github.com/cloudmesh-community/book/chapters

In case you see an error or like to make a contribution of your own section or chapter, you can do so in github via pull requests.

The easiest way to fix an error is to read the ePub and click on the cloud symbol in a heading where you see the error. This will bring you to an editable document in github. You can directly fix the error in the web browser and create there a pull request. Naturally, you need to be signed into github before you can edit and create a pull request.

As a result contributors and authors will be integrated automatically next time we compile the material. Thus even if you corrected a single spelling error, you will be acknowledged.

5.3 Contributors

Contributors are sorted by the first letter of their combined Firstname and Lastname and if not available by their github ID. Please, note that the authors are identified through git logs in addition to some contributors added by hand. The git repository from which this document is derived contains more than the documents included in this document. Thus not everyone in this list may have directly contributed to this document. However if you find someone missing that has contributed (they may not have used this particular git) please let us know. We will add you. The contributors that we are aware of include:

> *Anand Sriramulu, Ankita Rajendra Alshi, Anthony Duer, Arnav, Averill Cate, Jr, Bertolt Sobolik, Bo Feng, Brad Pope, Dave DeMeulenaere, De'Angelo Rutledge, Eliyah Ben Zayin, Eric Bower, Fugang Wang, Geoffrey C. Fox, Gerald Manipon, Gregor von Laszewski, Hyungro Lee, Ian Sims, IzoldaIU, Javier Diaz, Jeevan Reddy Rachepalli, Jonathan Branam, Juliette Zerick, Keith Hickman, Keli Fine, Mallik Challa, Mani Kagita, Miao Jiang, Mihir Shanishchara, Min Chen, Murali Cheruvu, Orly Esteban, Pulasthi Supun, Pulasthi Supun Wickramasinghe, Pulkit Maloo, Qianqian Tang, Ravinder Lambadi, Richa Rastogi, Ritesh Tandon, Saber Sheybani, Sachith Withana, Sandeep Kumar Khandelwal, Silvia Karim, Swarnima H. Sowani, Tharak Vangalapat, Tim Whitson, Tyler Balson, Vafa Andalibi, Vibhatha Abeykoon, Vineet Barshikar, Yu Luo, ahilgenkamp, aralshi, bfeng, brandonfischer99, btpope, garbeandy, harshadpitkar, himanshu3jul, hrbahramian, isims1, janumudvari, joshish-iu, juaco77, karankotz, keithhickman08, mallik3006, manjunathsivan, qianqian tang, rajni-cs, rirasto, shilpasingh21, swsachith, trawat87, tvangalapat, varunjoshi01, vineetb-gh, xianghang mi, zhengyili4321*

5.4 Creating the ePubs from source

In case you wish to create the ePub from source, we have included this section.

The creation of the book is based on bookmanager.

The easiest way is to use our docker container as described in Section 5.4.1.

5.4.1 Docker

We recommend the docker creation method for

- Ubuntu
- Windows 10
- macOSX

5.4.1.1 Using OSX

The easiest way to create a system that can compile the book on macOS, is to use a docker container. To do so you will need to first install docker on macOS while following the simple instructions at

- https://docs.docker.com/docker-for-mac/install/

Once you have docker installed, you can follow the instructions in Section 5.4.1.

5.4.1.2 Using the Docker Image

In case you have docker installed on your computer you can create ePubs with our docker image. To create that image by hand, we have included a simple makefile. Alternatively you can use our image from dockerhub if you like, it is based on ubuntu and uses our Dockerfile.

First, you need to download the repository:

```
$ git clone https://github.com/cloudmesh-community/book.git
cd book
```

To open an interactive shell into the image you say

```
$ make shell
```

The container image includes

- Python 3.7.4
- Pandoc 2.7.3
- pandoc-citeproc

Now you can skip to Section 5.4.4 and compile the book just as documented there.

Please note that we have not integrated pandoc-mermaid and pandoc-index at this time in our docker image. If you like to contribute them, please try it and make a pull request once you got them to work.

In case you want to create or recreate the image from our Dockerfile (which is likely not necessary, you can use the command

```
$ make image
```

5.4.2 Using the Native System

In case you like to use your native environment (which is typically faster than the container) you need to make sure you have an up to date environment.

Please note, that you must have at least Pandoc version 2.5 installed as earlier versions will not work. We recommend that you use pandoc version 2.7.3 or newer. We recommend that you use Python version 3.7.4 to run the scripts needed to assamble the document. However eralier version of Python 3 may also work, but are not tested. You can check the versions with

```
$ pandoc --version
$ python --version
```

5.4.3 Using Vagrant

In case you have installed vagrant on your computer which is available for macOS, Linux, and Windows 10, you can use our vagrant file to start up a virtual machine that has all software installed to create the ePub.

First, you need to download the repository:

```
$ git clone https://github.com/cloudmesh-community/book.git
```

```
$ cd book
```

Next you have to create the virtual machine with

```
$ vagrant up
```

You can loginto the VM with

```
$ vagrant ssh
```

The book folder will be mounted in the VM and you can can follow the instructions in Section 5.4.1.

5.4.4 Creating a Book

Once you have decided for one of the methods, you can create a book.

To create a book, you have to first check out the book source from github with if you have not yet done so (for example if you were to use the docker container method):

```
git clone git@github.com:cloudmesh-community/book.git
```

Books are organized in directories. We currently have created the following directories

```
./book/books/cloud/
./book/books/big-data-applications/
./book/books/pi
./book/books/writing
./book/books/222
./book/books/516
```

To compile a book go to the directory and make it. Let us assume you like to create the cloud book for cloud

```
$ git clone https://github.com/cloudmesh-community/book.git
$ cd book/books/cloud
$ make
```

To view it you say

```
$ make view
```

After you have done modifications, you need to do one of two things. In case you add new images you need to use

```
$ make
```

The structure of the books is maintained in the yaml file in the directory where you execute the make in. It typically has the form `NAMEOFDIR.yaml`. Simply do an ls in the directory to see its anme or inspect the Makefile. You can add new chapters to the yaml file, but discuss this first with Gregor. TYpicallly, we have for incomming or draft chapters a special `draft` book to make sure the integration is done smoothly first in the draft.

5.4.5 Publishing the Book to GitHub

> ⚠ *This task is only to be done by Gregor von Laszewski. You will not have to do this step.*

To publish the book say

```
$ make publish
```

5.4.6 Creating Drafts

Drafts are maintained in the draft folder

```
$ cd book/books/cloud
$ make
```

We recommend that you use the following tools to clean up your files.

- mdl - markdownlint to cleanup your markdown
- biber - to cleanup your bibtex file

We still only use bibtex and not biblatex, but can use biber for doing some verification. Once you have installed them, you can verify your documents with.

```
mdl filename.md
biber -V -tool filename.bib
```

Please remember that we have many thausends of refernces in our bib folder, so before you add a duplicate entry, please check in that forlder. An easy way to do this is to use jabref loading the bibfiles.

5.4.7 Creating a New Book

Let us assume you like to create a new book. The easiest way to start is to copy

from an existing book. However, make sure not to copy old files in dest. Let us assume you like to call the book gregor and you coppy from the python directory.

You have to do the following

```
$ cd book/books/python
$ make clean
$ cd ..
$ cp -r python gregor
$ cd ../gregor
$ mv python.yaml gregor.yaml
```

edit the Makefile and replace the NAME with gregor. make modifications to the table of contents in that yaml file and then compile with

```
$ make
```

5.4.8 Managing Images

In case you have added images to the book, they must be on the same level as your contribution, but in a directory called images. E.g.

```
./chapters/cloud/mydocument.md
./chapters/cloud/images/myimage.md
```

In the document the image is than refered to as

```
![My imaage caption](images/myimage.md){#fig:cloud-myimage}
```

The label `#fig:cloud-myimage` must be unique in all of the documents. While adding the directory cloud before the image name this is the case in our example.

5.4.9 Managing Refernces

Referncesa are all managed in bibtex format while using pandoc-crosreff to cite them. There are many examples of the different entry types available in the bib directory. DO not duplicte entries, instead reuse them. Make sure you have a unique and meaningful label.

5.5 Notation

The material here uses the following notation. This is especially helpful, if you

contribute content, so we keep the content consistent.

if you like to see the details on how to create them in the markdown documents, you will have to look at the file source while clicking on the cloud in the heading of the Notation section (Section 5.5). This will bring you to the markdown tex, but you will still have to look at the raw content to see the details.

☁ or  `![Github](images/github.png)`

> *If you click on the ☁ or  in a heading, you can go directly to the > document in github that contains the next content. This is > convenient to fix errors or make additions to the content. The cloud will be automatically added upon inclusion of a new markdown file that includes in its first line a section header.*

$

> *Content in bash is marked with verbatim text and a dollar sign*
>
> ```
> $ This is a bash text
> ```

[7]

> *References are indicated with a number and are included in the > reference chapter [7]. Use it in markdown with >* `[@las14cloudmeshmultiple]`*. References must be added to the* `refernces.bib` *file in BibTex format.*

🅾 or 

> *Chapters marked with this emoji are not yet complete or have some issue that we know about. These chapters need to be fixed. If you like to help us fixing this section, please let us know. Use it in markdown with* `:o2:` *or if you like to use the image with* `![No](images/no.png)`*.*

 REST 36:02

> *Example for a video with the* `![Video](images/video.png)` *emoji. Use it in*

markdown with `[![Video](images/video.png) REST 36:02](https://youtu.be/xjFuA6q5N_U)`

 Slides 10

Example for slides with the `![Presentation](images/presentation.png)` *emoji. These slides may or may not include audio.*

 Slides 10

Slides without any audio. They may be faster to download. Use it in markdown with `[![Presentation](images/presentation.png) Slides 10](TBD)`**.**

A set of learning objectives with the `![Learning](images/learning.png)` *emoji.*

A section is release when it is marked with this emoji in the syllabus. Usie it in markdown with `![Ok](images/ok.png)`**.**

Indicates opportunities for contributions. Usie it in markdown with `![Question](images/question.png)`**.**

Indicates sections that are worked on by contributors. Use it in markdown with `![Construction](images/construction.png)`**.**

Sections marked by the contributor with this emoji `![Smiley](images/smile.png)` *when they are ready to be reviewed.*

> *Sections that need modifications are indicated with this emoji* `![Comment](images/comment.png)`.

> *A warning that we need to look at in more detail* `![Warning](images/warning.png)`

> *Notes are indicated with a bulb* `![Idea](images/idea.png)`

Other emojis

> *Other emojis can be found at https://gist.github.com/rxaviers/7360908. However, note that emojis may not be viewable in other formats or on all platforms. We know that some emojis do not show in calibre, but they do show in macOS iBooks and MS Edge*

This is the list of emojis that canbe converted to PDF. So if you like a PDF, please limit your emojis to

`:cloud:` ☁ `:o2:` 🅾 `:relaxed:` ☺ `:sunny:` ☀ `:baseball:` ⚾ `:spades:` ♠ `:hearts:` ♥ `:clubs:` ♣ `:diamonds:` ♦ `:hotsprings:` ♨ `:warning:` ⚠ `:parking:` 🅿 `:a:` 🅰 `:b:` 🅱 `:recycle:` ♻ `:copyright:` © `:registered:` ® `:tm:` ™ `:bangbang:` ‼ `:interrobang:` ⁉ `:scissors:` ✂ `:phone:` ☎

5.5.1 Figures

Figures have a caption and can be refereed to in the ePub simple with a number. We show such a reference pointer while referring to Figure 1.

Figure 1: Figure example

Figures must be written in the md as

```
![Figure example](images/code.png){#fig:code-example width=1in}
```

Note that the text must be in one line and must not be broken up even if it is longer than 80 characters. You can refer to them with `@fig:code-example`. Please note in order for numbering to work figure references must include the `#fig:` followed by a unique identifier. Please note that identifiers must be really unique and that identifies such as `#fig:cloud` or similar simple identifiers are a poor choice and will likely not work. To check, please list all lines with an identifier such as.

```
$ grep -R "#fig:" chapters
```

and see if your identifier is truly unique.

5.5.2 Hyperlinks in the document

To create hyperlinks in the document other than images, we need to use proper markdown syntax in the source. This is achieved with a refernce for example in sections headers. Let us discuss the refernce header for this section, e.g. Notation. We have augmented the section header as follows:

```
# Notation {#sec:notation}
```

Now we can use the refernce in the text as follows:

```
In @sec:notation we explain ...
```

It will be rendered as: In Section 5.5 we explain …

5.5.3 Equations

Equations can be written as

```
$$a^2+b^2=c^2$${#eq:pythagoras}
```

and used in text:

$$a^2 + b^2 = c^2 \qquad (1)$$

It will render as: As we see in Equation 1.

The equation number is optional. Inline equations just use one dollar sign and do not need an equation number:

```
This is the Pythagoras theorem: $a^2+b^2=c^2$
```

Which renders as:

This is the Pythagoras theorem: $a^2 + b^2 = c^2$.

5.5.4 Tables

Tables can be placed in text as follows:

```
: Sample Data Table {#tbl:sample-table}

x   y   z
--- --- ---
1   2   3
4   5   42
```

As usual make sure the label is unique. When compiling it will result in an error if labels are not unique. Additionally there are several md table generators available on the internet and make creating table more efficient.

5.6 UPDATES

As all documents are managed in github, the list of updates is documented in the commit history at

- https://github.com/cloudmesh-community/book/commits/master

In case you do a lecture withus we recommend that you download a new version oof the ePub every week. This way you are typically staying up to date. You can check the commit history and identify if the version of the ePub is older than the committed version, if so we recommend that you download a new version.

> ⚠ *We typically will not make announcements to the class as the GitHub commit history is sufficient and you are responsible to monitor it as part of your class activities.*

6 REFERENCES

[1] G. von Laszewski, *Cloud computing*, Fall 2019. Bloomington, Indiana: Indiana University, 2019 [Online]. Available: https://laszewski.github.io/book/cloud/

[2] G. von Laszewski, *Cloud technologies*, Fall 2019. Bloomington, Indiana: Indiana University, 2019 [Online]. Available: https://cloudmesh-community.github.io/book/vonLaszewski-cloud-technologies.epub?raw=true

[3] G. von Laszewski, "Project format example." Aug-2019 [Online]. Available: https://github.com/cloudmesh-community/proceedings-fa18/tree/master/project-report

[4] G. von Laszewski, *Python for cloud computing*, Fall 2019. Bloomington, Indiana: Indiana University, 2019 [Online]. Available: https://laszewski.github.io/book/python/

[5] G. von Laszewski, *Linux for cloud computing*, Fall 2019. Bloomington, Indiana: Indiana University, 2019 [Online]. Available: https://laszewski.github.io/book/linux/

[6] G. von Laszewski, *Scientific writing with markdown*, Fall 2019. Bloomington, Indiana: Indiana University, 2019 [Online]. Available: https://laszewski.github.io/book/writing/

[7] G. von Laszewski, F. Wang, H. Lee, H. Chen, and G. C. Fox, "Accessing Multiple Clouds with Cloudmesh," in *Proceedings of the 2014 acm international workshop on software-defined ecosystems*, 2014, p. 8 [Online]. Available: https://github.com/cyberaide/paper-cloudmesh/raw/master/vonLaszewski-cloudmesh.pdf