Lesson 1.1: Introduction

UNIVERSITY OF MARYLAND

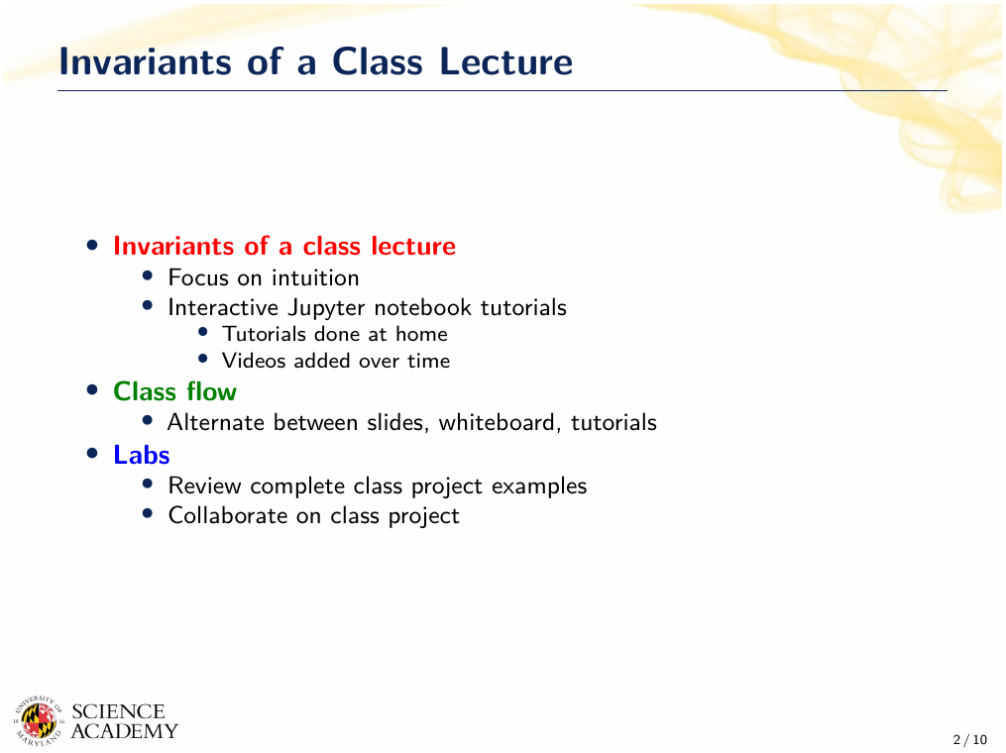
UMD DATA605: Big Data Systems

Lesson 1.1: Introduction

Instructor: Dr. GP Saggese, gsaggese@umd.edu

SCIENCE ACADEMY

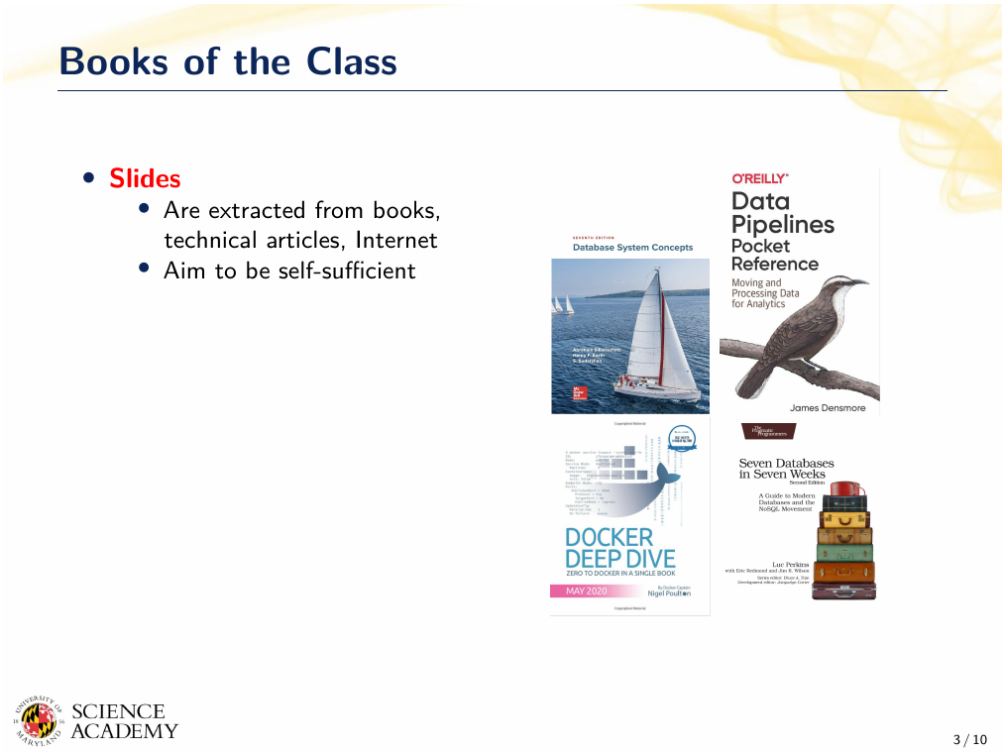
2 / 10: Invariants of a Class Lecture



- **Invariants of a class lecture**
  - *Focus on intuition*: The goal here is to ensure that students grasp the fundamental concepts and underlying principles of the subject matter. Instead of just memorizing facts, students are encouraged to understand the 'why' and 'how' behind the concepts.
  - *Interactive Jupyter notebook tutorials*: These are hands-on coding exercises that allow students to apply what they've learned in a practical setting. Jupyter notebooks are particularly useful because they combine code, visualizations, and narrative text in one place.
    - *Tutorials done at home*: Students are expected to complete these tutorials outside of class time, allowing them to learn at their own pace and revisit challenging sections as needed.
    - *Videos added over time*: Supplementary videos are provided to enhance understanding. These might include walkthroughs of the tutorials or additional explanations of complex topics.
- **Class flow**
  - The class alternates between different teaching methods: slides for structured information, whiteboard for dynamic explanations and problem-solving, and tutorials for hands-on practice. This varied approach caters to different learning styles and keeps the class engaging.
- **Labs**
  - *Review complete class project examples*: Students examine full project examples to understand how individual concepts come together in a comprehensive application. This helps in visualizing the end goal of their learning.
  - *Collaborate on class project*: Students work together on a class project, fostering team-

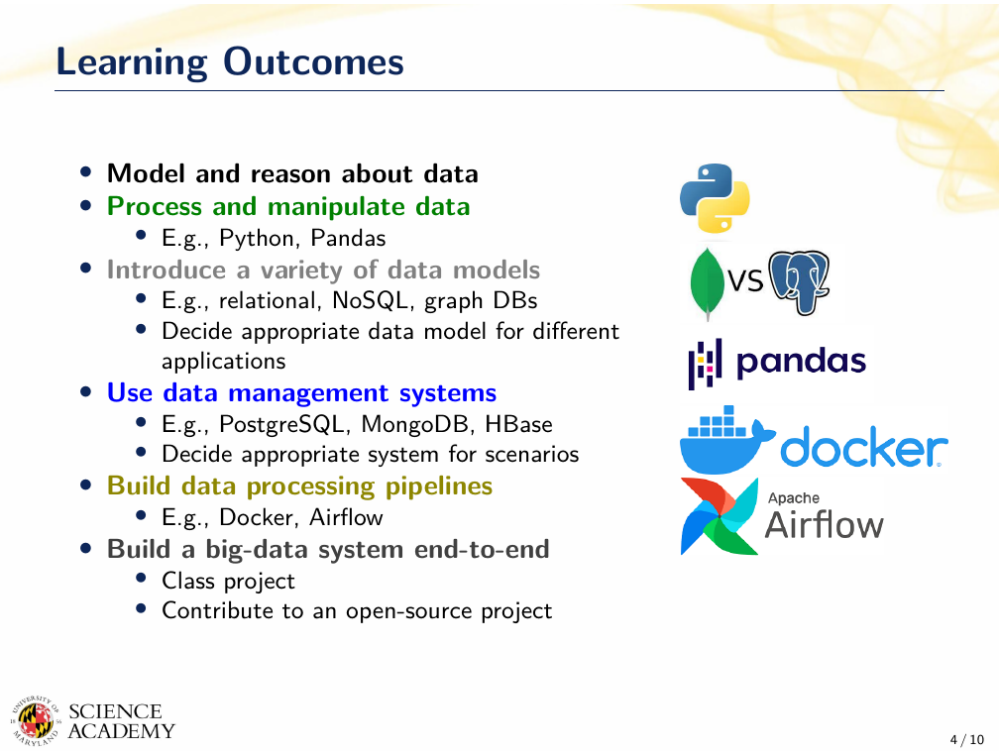
work and allowing them to learn from each other. Collaboration also mirrors real-world scenarios where teamwork is essential.

3 / 10: Books of the Class



- **Slides**
  - **Are extracted from books, technical articles, Internet**: The slides used in this class are compiled from a variety of sources, including textbooks, scholarly articles, and online resources. This approach ensures that the information presented is comprehensive and up-to-date. By drawing from multiple sources, the slides provide a well-rounded perspective on the topics covered in the course.
  - **Aim to be self-sufficient**: The slides are designed to be complete and understandable on their own. This means that even if you miss a lecture or need to review the material later, the slides should provide enough information to grasp the key concepts. They are structured to include definitions, explanations, and examples, making them a valuable resource for studying and revising the course material.
- **Images**: The images shown on the slide are likely covers or relevant pages from textbooks or articles used in the course. These visuals help to identify the primary sources of information and may also serve as a reference for further reading. By including these images, the slide emphasizes the importance of these texts in understanding the course content.

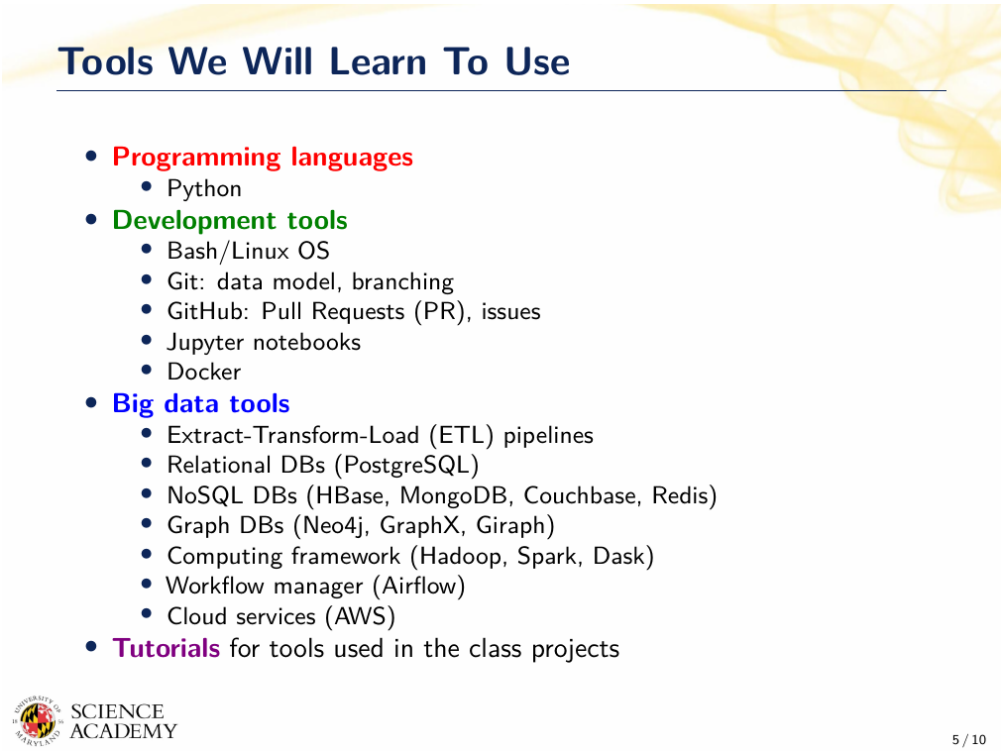
4 / 10: Learning Outcomes



- **Model and reason about data**
  - This outcome focuses on understanding how to create models that represent data accurately. It involves analyzing data to draw meaningful conclusions and make informed decisions. This skill is foundational in data science and machine learning, as it helps in understanding the underlying patterns and relationships within data.
- **Process and manipulate data**
  - This involves using tools like *Python* and *Pandas* to clean, transform, and prepare data for analysis. These tools are essential for handling large datasets efficiently, allowing you to perform operations such as filtering, aggregating, and reshaping data.
- **Introduce a variety of data models**
  - Understanding different data models, such as relational databases, NoSQL, and graph databases, is crucial. Each model has its strengths and is suited for specific types of applications. For example, relational databases are great for structured data, while NoSQL is better for unstructured data.
- **Use data management systems**
  - This involves learning how to use systems like *PostgreSQL*, *MongoDB*, and *HBase* to store and retrieve data. Choosing the right system depends on the specific needs of your application, such as the type of data, scalability requirements, and query complexity.
- **Build data processing pipelines**
  - Tools like *Docker* and *Airflow* are used to automate and manage data workflows. Building pipelines helps in efficiently processing data from raw input to final output, ensuring that data is consistently and reliably transformed and analyzed.
- **Build a big-data system end-to-end**
  - This is a practical application of the skills learned, where you will either work on a class

project or contribute to an open-source project. It involves designing, implementing, and deploying a complete big-data system, providing hands-on experience in managing large-scale data processing tasks.

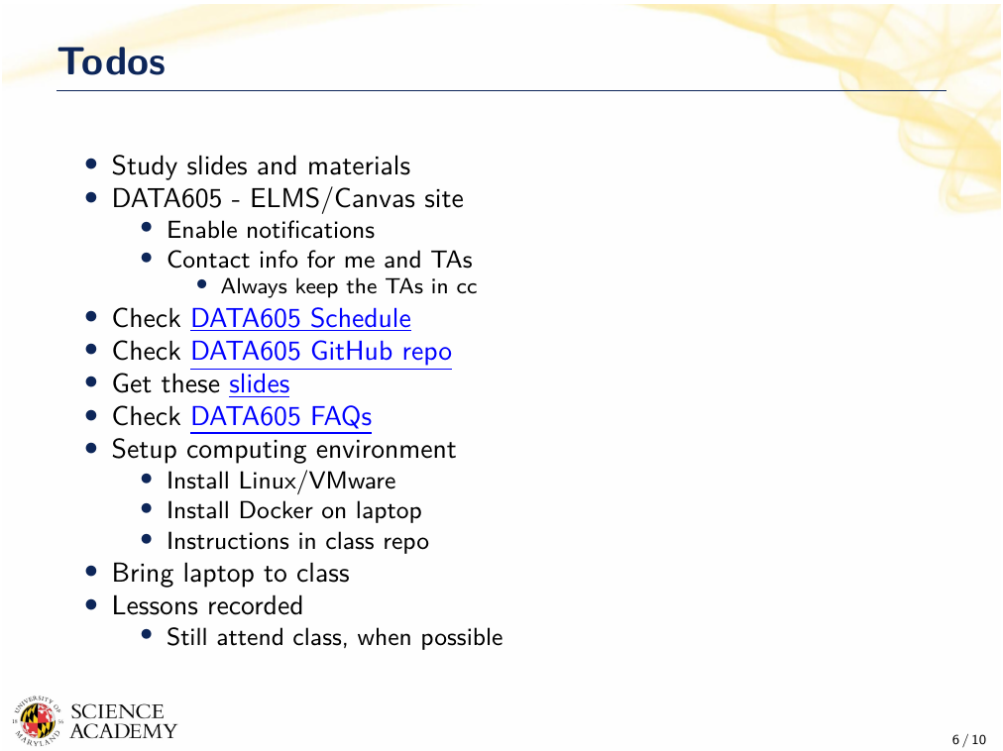
5 / 10: Tools We Will Learn To Use



- **Programming languages**
  - **Python**: We'll focus on Python because it's a versatile and widely-used language in data science and machine learning. It's known for its readability and has a rich ecosystem of libraries that make it ideal for data analysis and machine learning tasks.
- **Development tools**
  - **Bash/Linux OS**: Understanding the basics of Bash and Linux will help you navigate and manage files efficiently in a command-line environment, which is crucial for handling large datasets.
  - **Git**: We'll learn about Git's data model and branching to manage code versions effectively. This is essential for collaboration and maintaining a history of changes.
  - **GitHub**: We'll use GitHub for managing code repositories, focusing on Pull Requests (PR) and issues to facilitate collaboration and track project progress.
  - **Jupyter notebooks**: These are interactive tools that allow you to write and execute code in chunks, making it easier to document and share your data analysis and machine learning workflows.
  - **Docker**: Docker helps in creating consistent development environments by packaging applications and their dependencies into containers, ensuring that your code runs the same way everywhere.
- **Big data tools**
  - **Extract-Transform-Load (ETL) pipelines**: These are processes used to extract data from various sources, transform it into a suitable format, and load it into a database or data warehouse.
  - **Relational DBs (PostgreSQL)**: We'll explore PostgreSQL, a powerful open-source relational database, to manage structured data using SQL.

  - **NoSQL DBs (HBase, MongoDB, Couchbase, Redis)**: These databases are designed to handle unstructured data and provide flexibility in data storage and retrieval.
  - **Graph DBs (Neo4j, GraphX, Giraph)**: Graph databases are used to model and query data with complex relationships, which is useful in scenarios like social networks.
  - **Computing framework (Hadoop, Spark, Dask)**: These frameworks allow for distributed processing of large datasets across clusters of computers, making data processing faster and more efficient.
  - **Workflow manager (Airflow)**: Airflow helps in scheduling and monitoring workflows, ensuring that data processing tasks are executed in the correct order.
  - **Cloud services (AWS)**: We'll explore Amazon Web Services (AWS) for scalable cloud computing resources, which are essential for handling big data projects.
- **Tutorials** for tools used in the class projects: We'll provide tutorials to help you get hands-on experience with these tools, ensuring you can apply what you learn to real-world projects.

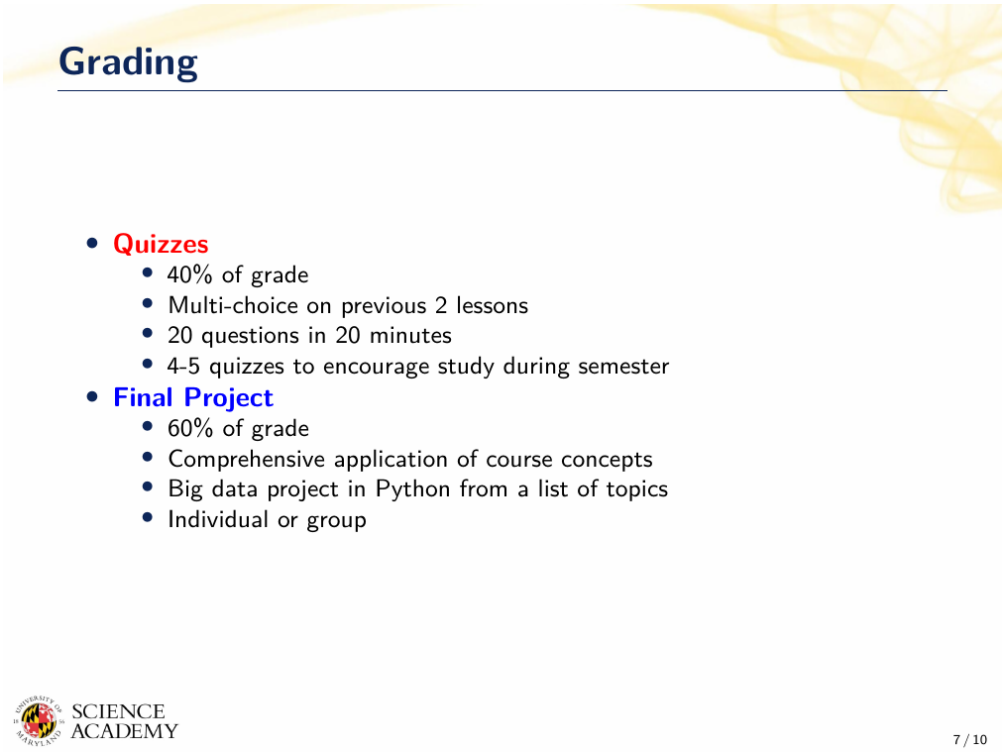
6 / 10: Todos



- **Study slides and materials**
  - It's important to review all the slides and materials provided for the course. This will help you understand the topics covered and prepare for discussions and assignments.
- **DATA605 - ELMS/Canvas site**
  - **Enable notifications**: Make sure you have notifications turned on so you don't miss any important updates or announcements.
  - **Contact info for me and TAs**: Keep the contact information for the instructor and teaching assistants handy. It's crucial to include TAs in your emails by keeping them in cc to ensure everyone is on the same page.
- **Check DATA605 Schedule**
  - The schedule is available on Google Docs. Regularly checking it will help you stay on track with deadlines and upcoming classes.
- **Check DATA605 GitHub repo**
  - The GitHub repository contains important resources and code examples. Familiarize yourself with its contents to aid your learning process.
- **Get these slides**
  - Access the slides from the GitHub link provided. They are essential for following along with lectures and understanding the course material.
- **Check DATA605 FAQs**
  - The FAQs document can answer common questions you might have about the course. It's a good first stop for any queries.
- **Setup computing environment**
  - **Install Linux/VMware**: Setting up a Linux environment or using VMware is necessary for running certain software and tools used in the course.

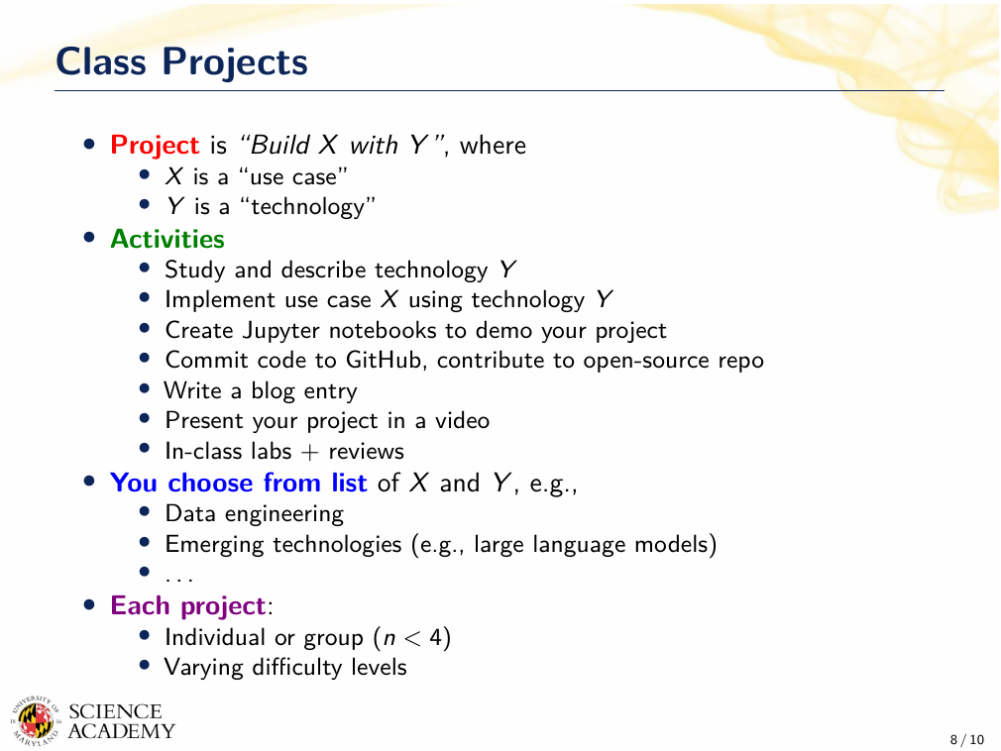
  - **Install Docker on laptop**: Docker is a tool that allows you to run applications in containers. Follow the instructions in the class repository to install it correctly.
- **Bring laptop to class**
  - Having your laptop in class is important for participating in hands-on activities and following along with coding exercises.
- **Lessons recorded**
  - While lessons are recorded for later viewing, attending class in person is beneficial for engaging with the material and asking questions in real-time.

7 / 10: Grading



- **Grading**
  - **Quizzes**
    - *40% of grade*: Quizzes make up a significant portion of your final grade, so it's important to take them seriously. They are designed to ensure you are keeping up with the course material.
    - *Multi-choice on previous 2 lessons*: Each quiz will cover content from the last two lessons, helping reinforce your understanding and retention of recent topics.
    - *20 questions in 20 minutes*: The quizzes are timed, with one minute per question, which means you need to be familiar with the material to answer quickly and accurately.
    - *4-5 quizzes to encourage study during semester*: Regular quizzes are spread throughout the semester to motivate consistent study habits and prevent last-minute cramming.
  - **Final Project**
    - *60% of grade*: The final project is the most substantial part of your grade, reflecting its importance in demonstrating your comprehensive understanding of the course.
    - *Comprehensive application of course concepts*: This project requires you to apply what you've learned throughout the course, showcasing your ability to integrate and utilize various concepts.
    - *Big data project in Python from a list of topics*: You'll choose a topic from a provided list, focusing on big data analysis using Python, which is a key skill in the field.
    - *Individual or group*: You have the flexibility to work alone or collaborate with peers, allowing you to choose the working style that best suits you.

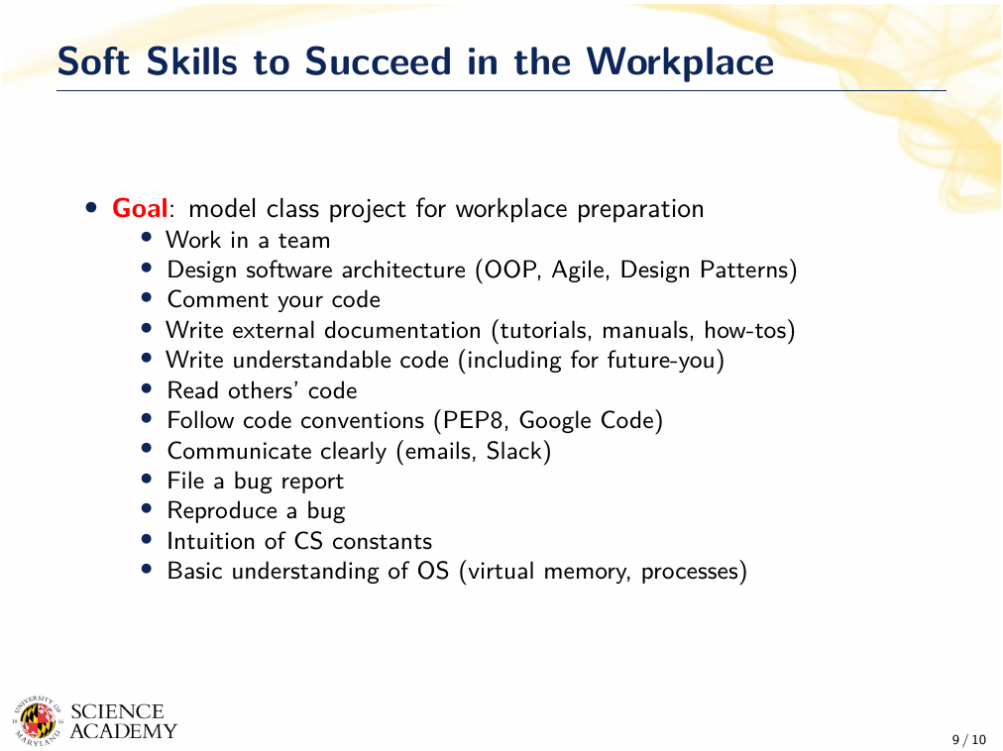
8 / 10: Class Projects



- **Class Projects**: The main goal of these projects is to combine a specific *use case* (represented by X) with a particular *technology* (represented by Y). This approach helps students apply theoretical knowledge to practical scenarios, enhancing their understanding and skills.
- **Activities**:
  - **Study and describe technology** Y: Begin by thoroughly understanding the technology you will use. This involves researching its features, capabilities, and limitations.
  - **Implement use case** X **using technology** Y: Apply the technology to solve a real-world problem or scenario, demonstrating how it can be effectively utilized.
  - **Create Jupyter notebooks to demo your project**: Use Jupyter notebooks to document your work, providing a clear and interactive way to showcase your project.
  - **Commit code to GitHub, contribute to open-source repo**: Share your code on GitHub, which not only helps in version control but also contributes to the open-source community.
  - **Write a blog entry**: Document your project journey, insights, and outcomes in a blog post to share your experience and learnings with a broader audience.
  - **Present your project in a video**: Create a video presentation to visually and verbally communicate your project, highlighting key aspects and results.
  - **In-class labs + reviews**: Participate in hands-on labs and peer reviews to refine your project and receive constructive feedback.
- **You choose from list** of X and Y: You have the flexibility to select from various use cases and technologies, such as data engineering or emerging technologies like large language models. This allows you to tailor the project to your interests and career goals.

- **Each project**:
  - **Individual or group ($n < 4$)**: Projects can be done individually or in small groups, fostering collaboration and teamwork.
  - **Varying difficulty levels**: Projects come with different levels of complexity, allowing you to challenge yourself according to your skill level and learning objectives.

9 / 10: Soft Skills to Succeed in the Workplace



- **Goal**: model class project for workplace preparation
  - *Work in a team*: Collaborating effectively with others is crucial in any workplace. It involves sharing ideas, dividing tasks, and supporting each other to achieve a common goal.
  - *Design software architecture (OOP, Agile, Design Patterns)*: Understanding these concepts helps in creating scalable and maintainable software. Object-Oriented Programming (OOP) is about organizing code into objects, Agile is a flexible project management approach, and Design Patterns are standard solutions to common problems.
  - *Comment your code*: Writing comments in your code helps others (and your future self) understand the purpose and functionality of your code.
  - *Write external documentation (tutorials, manuals, how-tos)*: Good documentation is essential for users to understand how to use your software effectively.
  - *Write understandable code (including for future-you)*: Code should be clear and easy to read, which helps in maintaining and updating it over time.
  - *Read others' code*: Being able to understand and learn from others' code is a valuable skill that can improve your own coding practices.
  - *Follow code conventions (PEP8, Google Code)*: Adhering to coding standards ensures consistency and readability across different projects and teams.
  - *Communicate clearly (emails, Slack)*: Clear communication is key to avoiding misunderstandings and ensuring everyone is on the same page.
  - *File a bug report*: Knowing how to document and report bugs accurately helps in resolving issues efficiently.
  - *Reproduce a bug*: Being able to replicate a bug is the first step in diagnosing and fixing it.

- *Intuition of CS constants*: Having a basic understanding of constants in computer science, like time complexity, helps in writing efficient code.
- *Basic understanding of OS (virtual memory, processes)*: Knowing how operating systems manage resources like memory and processes can help in optimizing software performance.

10 / 10: Yours Truly

Yours Truly

- **GP Saggese**
  - 2001-2006, PhD / Postdoc at the University of Illinois at Urbana-Champaign
  - LinkedIn
  - gsaggese@umd.edu
- **University of Maryland**:
  - 2023-, Lecturer for UMD DATA605: Big Data Systems
  - 2025-, Lecturer for UMD MSML610: Advanced Machine Learning
- **In the real-world**
  - Research scientist at NVIDIA, Synopys, Teza, Engineers' Gate
  - 3x AI and fin-tech startup founder (ZeroSoft, June, Causify AI)
  - 20+ academic papers, 2 US patents

SCIENCE ACADEMY

10 / 10

- **GP Saggese**
  - GP Saggese has a strong academic background, having completed a PhD and postdoctoral research at the University of Illinois at Urbana-Champaign between 2001 and 2006. This indicates a solid foundation in research and academia, particularly in fields related to machine learning and big data.
  - You can connect with GP Saggese on LinkedIn for professional networking or reach out via email at gsaggese@umd.edu for academic inquiries or collaborations.
- **University of Maryland**
  - Since 2023, GP Saggese has been a lecturer at the University of Maryland, teaching a course on Big Data Systems (UMD DATA605). This role highlights his expertise in handling and teaching about large-scale data systems.
  - Starting in 2025, he will also be teaching Advanced Machine Learning (UMD MSML610), which suggests a deep understanding of complex machine learning concepts and techniques.
- **In the real-world**
  - GP Saggese has practical experience as a research scientist at notable companies like NVIDIA, Synopsys, Teza, and Engineers' Gate. This experience bridges the gap between theoretical knowledge and practical application in industry settings.
  - He is an entrepreneur, having founded three AI and fintech startups: ZeroSoft, June, and Causify AI. This entrepreneurial experience demonstrates his ability to innovate and apply machine learning and big data concepts to real-world problems.
  - With over 20 academic papers and 2 US patents, GP Saggese has contributed significantly to the academic and technological community, showcasing his role as a thought leader in his field.