Insights from the Tech-in-Residence Corps:

Industry-informed Courses for NYC's High-Demand Tech Careers

November 2018

Insights from the Tech-in-Residence Corps:

Industry-informed Courses for NYC's High-Demand Tech Careers

November 2018

In June 2017, New York City Mayor Bill de Blasio announced ***a new goal of doubling the number of New Yorkers who earn tech bachelor's degrees from NYC public colleges by 2022***. This $20 million initiative – known as **"CUNY 2x Tech"** – includes investments in new Computer Science faculty lines, expanded academic and career advising for Computer Science students, and support for more internship, co-op, and work-study programs.

Among the critical components of CUNY 2x Tech is the ***Tech-In-Residence Corps*** **–** a new model of industry-academic collaboration that addresses the problems of departmental capacity, obstacles to adjuncting, and students' "applied knowledge" gaps by enabling more tech leaders to directly teach students the most in-demand skills needed for tech jobs in NYC.

The **Tech-in-Residence Corps** brings industry professionals from the NYC tech ecosystem into college classrooms to transfer their skillsets and applied knowledge directly to students and to collaborate with Computer Science faculty. One year since launch, the program consists of over 40 industry professionals representing 33 companies in NYC's tech ecosystem – including *Google, LinkedIn, Etsy, Bank of America, Bloomberg*, *StreetEasy* and more. The program has worked with the Computer Science departments at eight CUNY senior colleges to facilitate the delivery of 33 for-credit, advanced elective course sections reaching over 600 students. Courses taught by Corps members include:

- Agile Software Development
- Artificial Intelligence
- Big Data
- Cybersecurity
- Cyber Engineering
- Data Analytics
- Data Mining and Data Warehousing
- Data Warehousing for Analytics
- Digital Trends and Transformations
- Intro to Data Science
- Mobile Application Development
- Software Engineering
- Statistics for Data Science
- Web Development

Software Engineering:

Overview

Software engineers transform ideas and requirements into products for multiple platforms – for instance desktops, servers, mobile devices, and websites.

From November 2017 through October 2018, there were over 38,000 job postings from NYC employers seeking to hire software engineers, including **over 3,200 opportunities for individuals with 0-2 years of experience.**[1]

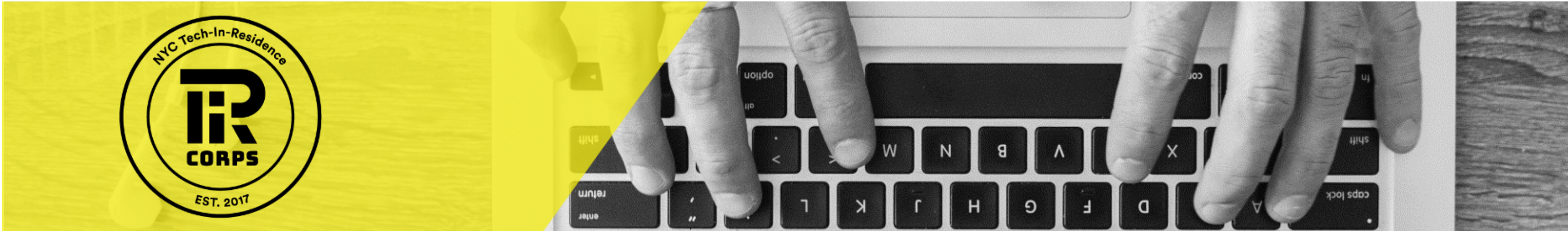
Courses taught by the full-time professional engineers of the Tech-in-Residence Corps integrate current relevant technologies and emphasize the "applied knowledge" skills that students will need as they enter the workforce.

Examples of such applied knowledge areas include Agile Development, Sprint and Scrum methodologies, modern version control systems such as Git and Github, and collaborative tools such as Slack and Codepen deemed critical for working in team-based professional environments. Courses also focus on applying foundational technical concepts – including algorithms and data structures – in settings that more accurately reflect the types of environments and challenges that graduates will face on the job.

To that end, of the numerous sections of Software Engineering courses that have been taught by Tech-in-Residence Corps Members, all universally incorporated **team-based** and **project-based** course design.

➢ **Additional Resource:** For additional information on the skills required for software engineering roles, review *NYC Industry Insights: Key Skills and Competencies Needed for In-Demand, Entry-Level Jobs in New York City* http://www.techtalentpipeline.nyc/nyc-industry-insights/.

[1] Source: Labor Insight Jobs (Burning Glass Technologies)



Software Engineering:

Sample Syllabus

City College of New York (CCNY)
Taught by Nikolai Avteniev, LinkedIn

***Nikolai Avteniev** has worked for LinkedIn since 2012 on several projects including: working with the Growth team to expand members' ability to import their address books, working with the API team to create APIs that allowed members to use LinkedIn in compliance with financial regulations, helping to build the content ingestion service, and more. As a member of the Oregon Data Center working group, Nikolai helped deploy the hundreds of services which power LinkedIn's next-generation data center. Currently, he's working with the video sharing team.*

Course Objectives

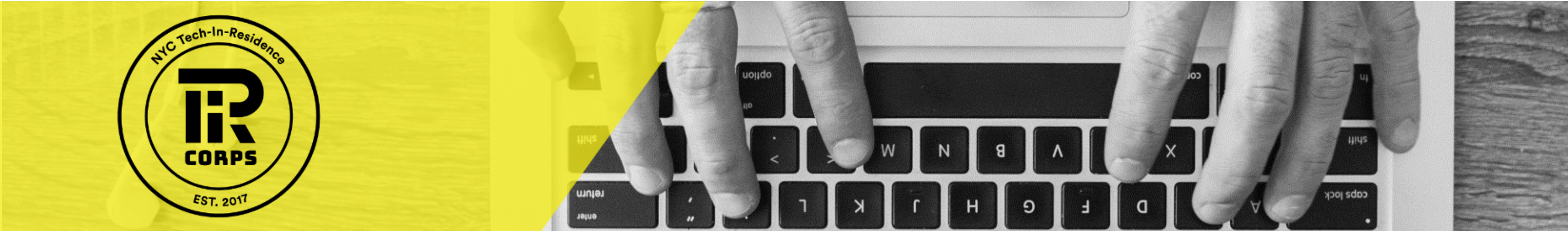
- Students will acquire knowledge of:
  - Agile software development
  - Automated testing techniques
  - Continuous integration and continuous deployment techniques
  - Collaborative planning and estimating
  - Collaborative implementation techniques
  - Agile software architecture techniques
  - The business responsibilities of the software engineer
  - Empirical software engineering practices
  - Industrial software engineering practices
  - User Experience Design (UX)

- Students will acquire ability to:
  - Work effectively as a part of an Agile team
  - Design and build large systems in iterative increments

Course Schedule

1	Introduction to Agile Software Development
2	Software Development is a Team Sport
3	Role of Customers and Users in Software Development
4	Planning and Estimating
5	Measuring Software Development, What and How?
6	Building Quality in Role of Test Automation in Software Development, Perpetual Beta, Continuous Integration and Delivery

7	Battling Complexity, Source Code Management, Continuous Refactoring, and Peer Code Review
8	Role of Architecture and Design in Agile Software Development
9	Iterating on Agile, Continuous Improvement
10	Seeing the Code, Monitoring and Observing your Systems in Production
11	Infrastructure as a Service, Agile Infrastructure on the Public Cloud
12	When Things Go Wrong, Handling Production Issues
13	Making Users Awesome, UX / HCI
14	Modern Agile, where do we go from here?



Software Engineering:

Sample Project Assignment

Hunter College
Taught by Alejandro Gonzalez, Jesse Greenberg, Sabeena Lalwani, Sean Laude, James Lin, Sid Wighe (AppNexus)

Course Project Deliverables

This is an outline of the deliverables you will complete over the course of the class. In the interests of simulating the experience of working in industry, we prefer to call these *deliverables* rather than *assignments*. These will be a mixture of presentations, written documentation, code artifacts, and/or actual running programs.

Regular Deliverables

Sprint Retro Notes

You should run a brief retro meeting after each sprint and take notes. These notes should be posted on your team Github repo.

User Stories

Project teams will write out their user stories and have them entered into the issue tracking system. These should be updated at least weekly.

One-Off Deliverables

Individual Project Pitches

Students will come up with ideas for projects and present them to the class.

Requirements

- Must be a novel idea (not already implemented somewhere else or be a major variation of an existing product or service)
- Some description of how it will be made, the technologies used, etc.
- Some statement of why or how this will be useful.

Example

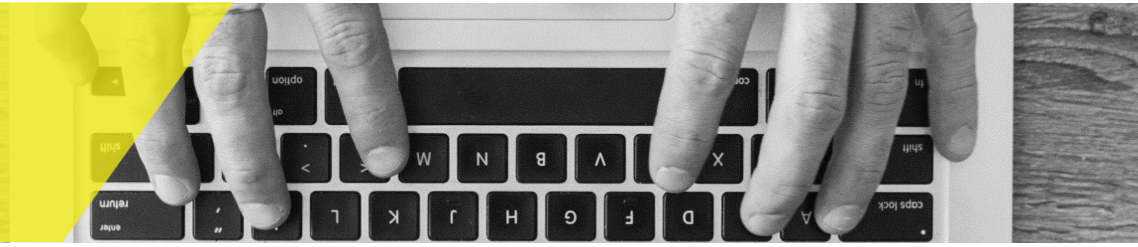
I would like to make Instragram for dinosaurs, e.g. *Dinogram*. This will be a mobile app that runs COBOL on the backend. I think this is useful because dinosaurs need a social media presence.

Team Product Proposals

Teams will be formed and their product proposals will be presented to the class

Requirements

- Must be a novel idea or team member's adaptation from the individual project pitches
- Adhere to the product definition section Product Thinking defining the target audience, problem, vision, strategy and goals.
- Some description of how it will be made, the technologies used, etc.



Presentations should consist of the following:

- Brief demo of the prototype (whatever you have "working" so far)
- Presentation and discussion of challenges that the team has encountered and overcome
- Presentation and discussion of challenges that the team has encountered and is still struggling with

Final Demo

Each product or application will be demonstrated to show all available features or capabilities.

Other Deliverables

UX storyboards and wireframes

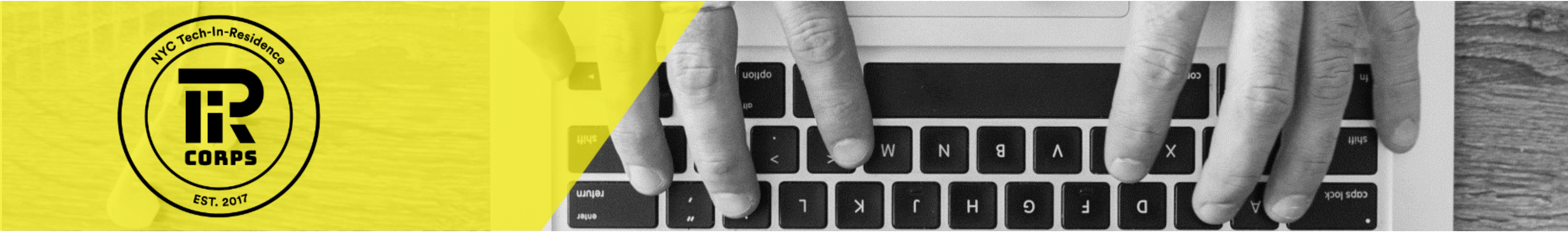
Each user story or set of related user stories will have UX storyboards or wireframes completed and entered into the issue tracking system

UI Prototype

A working user-interface application will be written and demonstrated

MVP Demo

Minimal viable product demonstrations will prove that the project application has its core features implemented fully.



Web Development:

Overview

Web Development continues to be one of the most popular jobs for recent graduates with a software engineering background. Employers in New York continue to seek out candidates who can design, develop, and maintain high-quality, responsive websites to support their business.

From November 2017 through October 2018, in addition to the high demand for software engineers, employers in NYC sought to hire over 6,600 web developers at all levels.[2] This is distinct from the number of software engineer postings that might include similar responsibilities.

There are, in fact, several different specialized tracks within the Web Development space – including front-end development, back-end development, full-stack development, UX/UI, etc. Nearly all of the course sections taught by Tech-in-Residence Corps Members – across all of these specialized tracks – have stressed the importance of **responsive design libraries** like Bootstrap and **the central importance of frameworks** such as AngularJS, React-Redux, Vue.js, Django, Flask, Express (Node.js), and more.

- ➢ **Additional Resource:** For additional industry insights on in-demand web development skills, access *Key Takeaways: TTP Web Development Summit*: http://www.techtalentpipeline.nyc/web-development-summit/.

[2] Source: Labor Insight Jobs (Burning Glass Technologies)

Web Development:

Sample Syllabus

Lehman College
Taught by Ross Dakin, Bank of America

***Ross Dakin** is a Senior Strategist of Global Technology & Operations at Bank of America Merrill Lynch. As a member of the Global Information Security R&D team, Ross helps identify new ways to protect the Bank and its customers from real and potential threats. Ross formerly advised the Microsoft Cities team on how to best utilize the company's resources for public good in NYC following the completion of his appointment as a Presidential Innovation Fellow in the Obama White House, where he worked on a number of national priorities through the lens of a private sector technologist.*

Course Objectives

By the end of this course, students will be well-versed in the fundamentals of web application development. Emphasis is placed on developing a foundation in plain HTML, CSS, and JavaScript, including a knowledge of their historical revisions and browser adoption of their latest versions. Building on this foundation, client-side JavaScript frameworks are introduced; students will learn the concepts, syntax, and benefits of jQuery then learn a more contemporary front-end JavaScript framework (e.g. Vue.js). The course will discuss back-end web application constructs such as databases, application servers, and network infrastructure, placing an emphasis on API-driven programming paradigms. By the completion of the course, students will be able to develop a client-side web application utilizing third-party data APIs and discuss the benefits of various architectures, design patterns, and elements of production-readiness.

Course Schedule

1	The Web, HTML
2	More HTML
3	URL anatomy, web colors, image types/considerations
4	CSS: selectors (element, class, ID); locations (inline, style block, external file)
5	Git, GitHub, Markdown, Slack, more CSS
6	Review HW-1, more CSS, page layout, form project groups, discuss Project 1
7	Project check-in, midterm review
8	Review HW-3
9	Floats, URL relativity, Grid, Flexbox, Midterm final questions
10	Midterm
11	Midterm recap, extra topics (database overview, etc.)
12	Project demos

13	Frameworks, design standards, style guides
14	Web product development
15	JavaScript Intro, dev tools (codepen, etc.)
16	Project demos, final review
F	Final exam

Web Development:

Sample Project Assignment

City College of New York (CCNY)
Taught by David Moon and Michelle Shu (Addepar)

Project
Over the next several weeks, in teams of 3-4, you will design and implement a fully functioning web application using React and Firebase.

Ideation
After forming your team, your first task is to decide on what to build. You have freedom in this decision, under the following constraints.

First, your application must persist user-specified data to Firebase. An anti-example is the keyboard drum kit page we implemented several weeks ago, which provides a fun acoustic experience but does not persist any data across sessions. A better example would be a drum beat interface that allows users to create, play, and save loops. If necessary, your application should perform validation on user input to prevent persisting any malformed data. If user input is invalid, then your application must indicate this clearly to the user.

Second, your application must provide non-trivial functionality beyond persisting and displaying data. As an anti-example, a basic to-do list that allows the user to create and complete to-do items in a static list does not provide sufficient functionality beyond persistence. A better example would be a to-do list that allows the user to group them by project and/or date, provides a drag-and-drop UI for re-organizing tasks, and supports searching/filtering on tasks.

Note: You can earn an extra 10 points (on top of 100) by integrating with a third-party API. Just as your application should provide functionality beyond simply displaying persisted data, you should aim to go beyond simply tacking on an unrelated display of third-party data. An anti-example is to add a list of recent Trump tweets to your to-do app. A better example is to extend your to-do list with a simple calendar view that juxtaposes your Google Calendar events alongside your tasks within each day.

Alternatively, you can earn an extra 10 points by utilizing a visualization library like D3.js. Once again, you should aim to go beyond a trivial usage of the library. Whatever you build with the library should constitute a fundamental piece of your application's user interface/experience.

You cannot earn more than 10 extra points.

If you are having trouble coming up with your own idea, you are welcome to implement the to-do list or drum beat loop ideas above. Depending on how large your group is, we may ask that you further extend the application with additional features. If you choose to implement the to-do list application, then you are required to come up with additional features that differ from what was proposed above—this is because I use the to-do list application to give you a sample tech spec in the next section. Another idea similar in

spirit to the to-do list is a personal expense manager in which you can record purchases and display summary statistics. This would be a good candidate for making use of the D3.js library.

Specification
Once you have decided upon what to build, your first assignment is to complete a design and technical specification for your application. Your specification should include the following pieces:

- mocks/sketches of your application in its various states
- a comprehensive list of user stories
- a complete React component hierarchy derived from your mocks
- the structure of persisted user-specified data
- if you utilize a third-party library, a description of the specific objects/classes/functions that you expect to leverage
- if you integrate with a third-party API, the API routes and the structure of the third-party data

You should treat this specification as a living document that you update as needed throughout the project timeline. (No need to update your initial mocks unless you undertake a drastic re-design.) From our perspective, we will be treating this specification as your documentation. How well-maintained your tech spec is will contribute to your project grade.

Process
Using Git & GitHub
Since you will be working in teams, it is essential that you leverage Git and GitHub effectively to organize and distribute your work. Each project team will be given a new GitHub repository in which to host their shared code.

As is the case with your personal repos, the `master` branch will be protected so that you cannot push changes directly to it. Each individual will contribute changes to the `master` branch through our familiar assignment submission procedure: check out the latest version of the GitHub repo to your local machine, create a new branch, commit and push your changes, and open a pull request to merge your branch into `master`. Review and approval by another team member will be required to merge.

In adopting this process, you will be practicing a standard workflow for collaboration in modern software development. This approach has important benefits: (1) developing on individual branches allows team members to work concurrently on the same codebase without stepping on each other’s toes, (2) establishing a formal process for merging into `master` upholds its sanctity, and (3) pull requests provide a good way to present the changes you made to your team members.

Stand-Up Meetings
Each week, we will schedule a 10-15 minute stand-up meeting with each team. In each meeting, every team member should be prepared to answer the following questions:

1. What did you accomplish in the past week?
2. Are you blocked on anything? If so, on what?
3. What will you be working on next?

You should be as specific as possible in your responses. When describing what you accomplished, mention some important details in your implementation and justify those choices if necessary, particularly if they affect the rest of the group. If you are blocked, come prepared with as detailed understanding of your issue as possible as well as specific questions. Think ahead about what you will be working on next, and discuss with team members how to distribute the next leg of work.

Your communication in these stand-up meetings will contribute to your overall project grade. This aspect does not depend on what you accomplished, but rather how clearly you communicate the details of your progress and any issues. It is expected that you will encounter unexpected issues.

Demo + Presentation

On 05/23, your group will give a final demo and slide-based presentation of your project. The demo and presentation together should take 15-20 minutes.

Think of your demo as an elevator pitch to potential investors. You should present a cohesive narrative of how the user would engage with your application, motivating and demonstrating its key features.

Your presentation will then dive into the details of implementation. Present the overall hierarchy of your React components, the data flow between them, and the structure of the data persisted to Firebase.

Reflect on your implementation experience and highlight any challenges you encountered. Describe what the problem was, how you addressed the issue, and what you took away from the experience once you resolved the issue.

Finally, your presentation should conclude with a retrospective on the software development process. How did you coordinate with and distribute work across your team? Which tools and workflows did you find useful, and which did you find to be burdensome? When did your plans or communication go awry? What would you do differently next time? What would you repeat? Be as specific as possible, citing concrete details and/or anecdotes.

Milestones

You should aim to meet the following milestones:

- Week 1 (04/25)
  - Tech spec complete
- Week 2 (05/02)
  - Tech spec revised based on feedback
  - React components can render (not necessarily styled) (similar to Lab 5)
  - React Router is set up
- Week 3 (05/09)
  - React component props and state are implemented
  - Data flows up and down the component hiearchy
- Week 4 (05/16)
  - Application can persist user-specified data to Firebase
  - Integration with third-party APIs (if any) is complete
- Week 5 (05/23):
  - Application is styled and polished
  - Demo + presentation are prepared

These milestones are a rough guideline – they may not make sense for your particular application. Let us know if you feel that some of the milestones above need to be rearranged for your application. For example, if your application is particularly UI-heavy with very simple persisted data, then some of the work that is due for Weeks 2 and 3 could be shifted a week back. However, you must let us know at least a week in advance before shifting any milestone. If we hit Week 1 and you have not talked with us about changing the work due for Week 2, then it is expected that you complete the Week 2 milestone described above. Only the code on `master` in your GitHub repo will be considered submitted each week.

Grading Rubric

- Tech spec (15 points)
    - Does your spec include mocks/sketches of your application in all of its possible states?
    - Does your spec include a comprehensive list of user stories?
    - Does your spec include a complete React component hiearchy?
    - Does your spec include the JSON structure of your persisted data?
    - If you utilize a third-party library, does your spec provide a description of the specific objects/classes/functions you will leverage?
    - If you intend on incorporating third-party data, does your spec include the JSON structure of the data and cite the API documentation?

- Code (50 points)
    - Does your code successfully implement each of your user stories?
    - Have you resolved error messages in the Chrome console?
    - Is your application reasonably styled and easy to use?
    - Are your components and any significant functions documented?
    - Does your code persist user-specified data?
    - Does your code provide non-trivial functionality beyond persisting and displaying data?

- Process (25 points)
    - Is your tech spec maintained and updated as the project progresses?
    - Are you creating Git branches and using GitHub pull requests to merge your work into the master branch?
    - Are you planning ahead and distributing work evenly among your teammates (as measured by number and size of pull requests)?
    - Are you meeting your weekly milestones?
    - Are you prepared for your stand-up meetings? Are you clearly communicating what you worked on, if and on what you are blocked, and what you will be working on next?

- Presentation (10 points)

Data-related Careers:

Overview

Demand for data professionals in New York City has grown dramatically over the past five years. From October 2013 through November 2018, the number of job postings for data analysts, data scientists, and data engineers grew by 74%, 685%, and 927%, respectively[3] – enormous year-over-year increases that far outpace the average growth rate for most jobs.[4] While the total numbers of jobs in each of those categories, at approximately 3,800, 2,400, and 1,540, respectively, are still much less than the total number of postings for software engineers, demand for these jobs has grown steadily over the past five years and is likely to continue growing. Moreover, when including closely related titles that require a core data skillset, such as marketing analysts, operations analysts, and business analysts, there are thousands more jobs that fall into this group of careers.

While there are a wide variety of roles that require data proficiency, they can be broadly grouped into three main types of jobs:

- **Data Analysts** glean insights from data to help businesses make good decisions. To do this, data analysts must be able to set up a framework for analysis, ask the right questions to arrive at a relevant answer, clean up datasets to facilitate analysis, use a variety of tools and methods to identify trends, and then present these findings clearly to technical and non-technical stakeholders in a business environment.

- **Data Scientists** take the work of data analysts steps further. Data Scientists also glean insights from data to help businesses make good decisions but are typically expected to be able to design and conduct complex, sophisticated analyses using advanced statistical, programming, modeling, and other skills to produce more powerful insights.

- **Data Engineers** are responsible for building the frameworks and infrastructure to capture and store the data that Analysts and Scientists use. Data Engineers help to ensure that data is stored efficiently, securely, and in a way that will make it easy for others to retrieve and use.

The Tech-in-Residence Corps has run numerous data-oriented courses including: Intro to Data Science, Data Analytics, Big Data, Data Warehousing for Analytics, Data Mining and Data Warehousing, and Statistics for Data Science.

➢ **Additional Resource:** For further information on skills required for data-related careers, explore *Key Takeaways: TTP Data Summit*: http://www.techtalentpipeline.nyc/data-summit/

[3] Source: Labor Insight Jobs (Burning Glass Technologies)

[4] Per U.S. Department of Labor, Bureau of Labor Statistics: "The [average] projected percent change in employment from 2016 to 2026…for all occupations is 7 percent."

Data Analysis

Sample Syllabus: Data Warehousing for Analytics

Baruch College
Taught by Royce Kok, EY

Royce Kok *is a senior consultant in the Technology Advisory Program at EY. Her clients are primarily in financial services and she works around financial models used for Capital Planning and Treasury. Royce's project experience includes data sourcing for model automation, model data lineage and dependencies.*

Course Description

This advanced course will provide students with an in-depth understanding of the design and implementation of database warehousing and analytics database systems. Specific topics include data warehouse modeling and architecture, the ETL process, administration, security, column-store, streaming and NoSQL databases, and complex event processing. Students develop a complete data warehouse system including implementation of a business intelligence suite.

Topics include:

- Brief Review of Relational Design and SQL
- Data Warehouse modeling and Architecture
- Extraction, Transformation and Loading (ETL)
- Data Warehouse Technical Architectures
- NoSQL Databases
- Big Data Processing: Hadoop/Spark
- Commercial and Open Source BI Tools
- BI Application Design and Development

Course Objectives

Upon successful completion of this course, students will be able to:

- Translate business needs and drivers into IT requirements for business intelligence systems
- Use the supporting technologies and data models for business intelligence including the process of and techniques for transforming business transaction data into appropriate analytic structures
- Explore state-of-the-art solutions for building and managing large data warehouse systems
- Discuss appropriate modeling approaches for a variety of industry specific requirements such as healthcare, banking, insurance, on-line advertising, and others.
- Develop a complete business intelligence system in a team setting using all of the tools and techniques presented during the course.

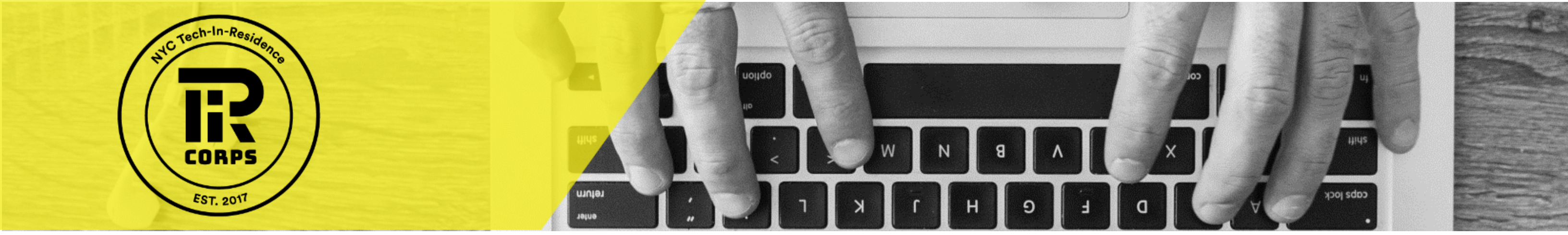
Upon successful completion of this course, students will have advanced skills to effectively design, develop, implement and manage medium to large-scale data warehouse systems.

<u>These advanced skills include:</u>

- Technology Literacy: Students will master technologies used to develop and deploy data warehouses and analytics systems.
- Knowledge Integration: Students will be able to analyze business requirements across multiple industries and address these requirements with appropriate data warehousing and analytics technologies.
- Written communication: Students will analyze a business and develop and write a business analytics proposal that will be implemented during the semester.
- Oral communication: Students will present their business analytics solution
- Teamwork and Leadership: Students will work in groups to analyze a business and develop and write a business analytics proposal that will be implemented during the semester.
- Ethical Awareness: Students will discuss issues of privacy, customer data collection and management, energy use by data centers, and ethical concerns when collecting, analyzing and presenting analytical data.

Course Schedule

1	Course Introduction and Review of E-R Model Relational Model and SQL
2	Data Warehouse Project Planning
3	Data Warehouse Architecture
4	Dimensional Modeling
5	Dimensional Modeling (Continued)
6	Extraction, Transformation and Loading (ETL) and ELT *Topic 1 – ETL Subsystems*
7	Extraction, Transformation and Loading (ETL) and ELT (Cont.) *Topic 2 – ETL System Consideration*
8	Extraction, Transformation and Loading (ETL) and ELT (Cont.) *Topic 3 – ETL Architecture*
9	Midterm exam
10	Data Warehouse Technical Architecture
11	Clustering and Distributed DBMS CAP Theorem
12	NoSQL Databases
13	Big Data Processing Architectures HADOOP and Spark
14	Web Applications Integration, XML and semi-structured data analytics
15	BI Application Design and Development
16	Final Exam Review



Data Analysis

Sample Project Assignment: Data Warehousing for Analytics

Baruch College
Taught by Royce Kok (EY)

Semester Project

The purpose of the semester-long project is to give students the experience of developing a working data warehouse using a commercial database management system and development tools. Students should form groups of 3 to 4 members per group.

Groups are encouraged to use open source data such as Opendata.gov, NYC Open Data, Kaggle and many others.

Stages of the Project

1. First, groups should submit for approval their project ideas. This proposal should include:

   a) A separate cover page indicating the title of your project (include full name and e-mail address of all members), the course number and course section.
   b) A narrative description of the business used for the data warehouse being created. This should also include a description of the problem or opportunity being addressed.
   c) Identification of the information needs – what data sources and information would help solve the problem or allow one to take advantage of the opportunity.
   d) Initial list of dimensions and facts that have been identified. This should come naturally from the above discussions. For a group project, I would expect a project to contain at least 5 dimensions and 2 fact tables.

   Informal discussions with the professor can help to refine the project and proposal.
   Students should not continue working on the project unless it has been approved by the instructor.

2. Groups should design the data warehouse schema by defining dimensions and fact tables. A diagram should be created to document this step.

3. Groups should then identify data sources and create ETL plans for extracting and loading this data into the data warehouse schema. Tools such as Pentaho Data Integration (Kettle) can be used. Take a screen picture of each major step and add a short note describing what is happening in this stage of the ETL process.

4. Once the ETL Process has been specified groups should populate a data warehouse schema with sample data.

5. Groups should then create a basic “Dashboard” application that displays at incorporates at least 3 different data representations such as graphs, maps, heatmaps, charts, tabular/cross-tab reports, etc. This may be implemented in any analytics package such as Oracle BI, Pentaho, Tableau, etc.

Project Milestones

- September 27, 2018: Groups are formed and a draft of projects ideas sent to professor.
- October 11, 2018: KPI’s and Data sources identified. Dimensional modeling started. Draft dimensional model sent to professor.
- November 1, 2018: Dimensional model finalized. ETL programming started. Final dimensional model sent to professor.
- November 20, 2018: ETL Programming completed. Dashboard programming started.
- November 29, 2018: Final project due.
- December 6, 2018 & December 11, 2018: Final project presentation

Project Deliverables

Each group should produce a document (MS Word) that includes all of the following:

- A separate cover page listing the course, group member’s names and project title.
- An introduction section similar to the proposal section including the narrative description of the business used for the data warehouse being created and description of the source data.
- The dimensional model diagram.
- A description and screen pictures of the ETL processes.
- Screen shots and brief descriptions of the final schema that the business analytics tools are working with.
- Screen shots and descriptions of the analytics (at least 3) on the dashboard application developed based on the data warehouse data.
- A narrative conclusion section that describes:
  - The group’s experience with the project (which steps were the most difficult? Which were the easiest? What did you learn that you did not imagine you would have? If you had to do it all over again, what would you have done differently?)
  - If the proposed benefits can be realized by the new system.
  - Any final comments and conclusions
- All of the above materials should be arranged in ONE MS Word document to be submitted electronically.

Data Science

Sample Syllabus: Intro to Data Science

City College of New York (CCNY)
Grant Long, StreetEasy

Grant Long *is head of the research team at StreetEasy and senior economist with the Zillow Group. Grant leads a team of analysts and data scientists dedicated to publishing research reports, producing data products, and working with real estate professionals, journalists, policy makers, and other stakeholders to better understand economic trends in New York City.*

Course Objectives

This course consists of a survey of analytical tools and concepts in data science, with goal of equipping students with an understanding of the best practices used by professional data scientists and analysts in top companies in technology, finance, and media. The course begins with an overview of fundamentals in data handling and exploratory data analysis, followed by an introduction to core concepts in statistical modeling and machine learning, and concludes with a brief introduction advanced concepts in data science. Students will work with a wide variety of real world data sets throughout the course in order to gain hands on experience. Emphasis will be placed on frequent practice through writing and reviewing code each week. In addition, students will be assigned and expected to discuss short reading assignments ranging from academic reviews of popular topics in analytics as well as data science and engineering blog posts from companies such as Airbnb, Facebook, and Spotify. Tasks and readings will aim to demystify the work of data teams in the real world and familiarize students with the concepts and resources needed to secure and succeed in analytical roles.

Upon successful completion of this course, students will be able to:

- Explain the key steps in a data science project
- Apply Python to load, clean, and process data sets
- Identify key elements of and patterns in a data set using computational analysis and statistical methods
- Explain and visualize empirical findings using with Python and other resources
- Explain fundamental principles of machine learning
- Apply predictive algorithms to a data set
- Work effectively in a team dedicated to analyzing data

Course Schedule

1	What is Data Science and Why Does It Matter?
2	Data Exploration 1: Loading, Summarizing, and Visualizing Data
3	Data Exploration 2: Dirty Data
4	Data Exploration 3: Storytelling and Statistics

5	Project Teams Formed; Models 1: Intro to Regression and Classification
6	Project Proposals Due; Models 2: Regularization, Variance/Bias
7	Midterm Exam; Machine Learning 1: Trees, Feature Selection
8	First Project Update; Machine Learning 2: Ensemble Models, Evaluation
9	Machine Learning 3: Bayes Rule and Bag of Words Methods for Text
10	Machine Learning 4: Unsupervised Learning
11	Second Project Update; Machine Learning 5: Bayesian Analysis and Scalable Data Science
12-13	TBD. Options: Big Data and the Cloud, Intro to Deep Learning, Recommender Systems, Advanced NLP
14	Course Review, Careers in Data, Data Ethics

Data Science

Sample Project Assignment: Intro to Data Science

City College of New York (CCNY)
Taught by Grant Long (StreetEasy)

Project Goal

The goal of the CSC 599.70 Course Project is to give students the opportunity to explore and analyze one or more data sets of their choosing with the goal of telling a compelling narrative using data. Through the project, students should:

- Apply Python to load, clean, and process data sets.
- Identify key patterns in a data set using computational analysis and statistical methods.
- Apply principles of statistical modeling and machine learning to data.
- Effectively explain, visualize, and communicate empirical findings.
- Demonstrate effective team collaboration.

The final product for the project should include data visualization, modeling – through regression, classification, or unsupervised learning – and a written and verbal presentation of findings that conveys the broader importance of the findings beyond the analytical exercise. In other words, students should be prepared not only to demonstrate their expertise and creativity in analyzing data, but also convince an external audience that their findings are important.

Contact & Information

For questions, comments, or requests for more information, please contact:
techtalentpipeline@sbs.nyc.gov

For more information on the Tech-in-Residence Corps courses and colleges, please visit:
http://cuny.edu/tech-in-residence-corps

For more information about joining the Tech-in-Residence Corps as an industry professional, please visit:
http://www.techtalentpipeline.nyc/tech-in-residence-corps/

The Tech-in-Residence Corps is a collaboration between NYC companies, the City University of New York, and the NYC Tech Talent Pipeline at the NYC Department of Small Business Services.

WEBSITE | techtalentpipeline.nyc

TWITTER | @NYCTechTalent

MEDIUM | medium.com/@nyctechtalent

NYC
Tech Talent
Pipeline

CUNY
THE GREATEST
URBAN UNIVERSITY
IN THE WORLD

NYC
Small Business
Services

careers
businesses
neighborhoods