THE TECH ACADEMY

MID-LEVEL DEVELOPER AND SOFTWARE ARCHITECT COURSE

April 18, 2019

TO: Graduates of The Tech Academy

PURPOSE: To train junior developers in the principles and practical application of software architecture, while simultaneously bringing their dev skill levels up to that of a mid-level developer (between the skill level of junior and senior – which typically takes about 3-5 years experience).

DESCRIPTION: This course was developed by Erik Gross and Jack Stanley (Co-Founders of The Tech Academy). It consists of external (not created by The Tech Academy) content. Steps include: reading books, completing online courses, reading articles, completing coding assignments, delivering whiteboard presentations and more. Just like when training to be an entry-level developer on our boot camps, "you don't know what you don't know" applies to moving from a junior to a middle skill level, and so we've laid out this path. This course includes all required knowledge that a mid-level developer is expected to have and trains one in software architecture.

INSTRUCTIONS: The steps on this course are done in sequence. Students are expected to complete each step in full prior to moving onto the next. The items have been placed in a specific order and skipping around will reduce the value and effectiveness of this course. Any words one doesn't understand should be cleared up as they're encountered. Do not go past any terms you do not fully understand. Definitions for common technical terms can be found in The Tech Academy's *Computer and Technology Basics Dictionary*. There are a series of practical assignments throughout this course. You are to use initiative and creativity in doing these assignments. Utilize all the skills you learned in the boot camp in terms of researching data and problem-solving.

TECH ACADEMY: This course is completed without any interaction with Tech Academy employees. Because they've already graduated a boot camp, students complete this without assistance from the school.

NOTE: You will be challenged on this course, possibly more so than you've ever experienced as a developer. Working yourself through the subject matter, though difficult, is a required element to truly becoming a mid-level software developer. There is a lot to learn, and you will find yourself needing to look things up yourself as you go, in order to understand and apply. As a graduate of our boot camp, you have learned how to self-teach and debug yourself, and these skills will now be put to the test. It is recommended that you find another developer (such as a friend or co-worker) to assist you through bugs you run into.

PRESENTATIONS: There are some whiteboard presentations on this course wherein you will explain concepts to another while utilizing a whiteboard to draw out data as needed. As a mid-level developer, you will be required to give presentations to groups of people and so practicing this skill is necessary to prepare you for the future.

If you don't have a whiteboard or another individual to deliver the presentation to in person, you can use an online presentation service (like a digital whiteboard and an online call). Like any other skill, comfort in public speaking can be obtained through practice.

ESSAYS: You will write some essays on this course but aren't required to submit them to anyone. The purpose of writing these is to simply ensure that you can describe the concepts covered in the essays. Your code and assignments don't need to be submitted to anyone either.

COMPLETION TIME: It is estimated that the average student will take over 200-250 hours of study time to complete this. This time estimate changes based on skill and experience level.

WRITING CODE: On this course you will study articles and videos that show code being written. *It is important that you write and execute all the coding depicted* – don't just view it and move on. By actually *doing* the actions covered in the content of this course, you'll have a higher retention of information and a greater increase in skill.

STUDY MATERIALS: You will need certain books and online training resources in order to complete this training program:

1. A membership to Pluralsight (www.pluralsight.com)
2. A Microsoft Azure subscription (you should be able to get a free trial if you don't already have one).
3. The Tech Academy's *Learn Coding Basics in Hours with Python* book.
4. The book *Code Simplicity: The Fundamentals of Software* by Max Kanat-Alexander. There is a Kindle version of this book.
5. The following books from the "Succinctly" series by Syncfusion, available at https://www.syncfusion.com/ebooks (Note: In order to download these books you will have to create an account with Syncfusion):
    a. *Public Speaking for Geeks Succinctly*
    b. *Docker Succinctly*
    c. *ASP.NET Core Succinctly*
    d. *Visual Studio 2017 Succinctly*
    e. *SQL Series Succinctly*
    f. *SQL Server for C# Developers Succinctly*
    g. *.NET Core Succinctly*
    h. *SOLID Principles Succinctly*
    i. *Object-Oriented Programming in C# Succinctly*

j. *ASP.NET Multitenant Applications Succinctly*
k. *Regular Expressions Succinctly*
l. *Localization for .NET Succinctly*
m. *Unit Testing Succinctly*
n. *Data Structures Succinctly Part 1*
o. *Data Structures Succinctly Part 2*
p. *HTTP Succinctly*

6. The book *The Complete Software Developer's Career Guide* by John Sonmez.

7. The book*Soft Skills: The Software Developer's Life Manual* by John Sonmez.

GITHUB: It is highly recommended that you add every coding project contained on this course to your GitHub account. An active GitHub account that displays a wide array of skills is one of the best ways to move up in the industry.

(OPTIONAL) SECTION ONE
C# FUNDAMENTALS

NOTE: The following steps are optional and are only included for students not trained in C#. If you already know the basics of C#, you can skip ahead to SECTION TWO.

1. Complete the C# Fundamentals with Visual Studio 2015 Course on Pluralsight.

2. Complete the C# Equality and Comparisons Course on Pluralsight.

3. Complete the C# Collections Fundamentals Course on Pluralsight.

4. Complete the C# Interfaces Course on Pluralsight.

5. Complete the C# Best Practices: Collections and Generics Course on Pluralsight.

6. Complete the C# Events, Delegates and Lambdas Course on Pluralsight.

7. Complete the Practical LINQ Course on Pluralsight.

8. Complete the Object-Oriented Programming Fundamentals in C# Course on Pluralsight.

9. Complete the Create an ASP.NET MVC 5 App with Facebook, Twitter, LinkedIn and Google OAuth2 Sign-on (C#) project found here in full.

SECTION TWO
COMPUTER SCIENCE FUNDAMENTALS

1. Read the first two Chapters of the book *Learn Coding Basics in Hours with Python* (through to the end of the chapter covering the five elements of a program).

2. ASSIGNMENT: Find a friend or associate (ideally someone with little to no technical skill) and deliver a short whiteboard presentation on how computers work. Get feedback from them. Then find another person and repeat the presentation (applying anything learned from the previous experience).

3. ASSIGNMENT: Find a friend or associate (ideally someone with little to no technical skill) and deliver a short whiteboard presentation on the basics of computer programming. Get feedback from them. Then find another person and repeat the presentation (applying anything learned from the previous experience).

4. Watch this video in full: Four Semesters of Computer Science in 5 Hours.

5. ESSAY: What is Big-O notation? How could you explain it to a person who isn't a technology worker? How can you apply it on the job as a programmer?

6. Read the book *Data Structures Succinctly Part 1* from Syncfusion. As you read, implement each data structure in a C# console application.

7. Read the book *Data Structures Succinctly Part 2* from Syncfusion. As you read, implement each data structure in a C# console application.

8. Read the book *SOLID Principles Succinctly* from Syncfusion.

9. ESSAY: How have you seen each of the SOLID principles successfully put into practice? How have you seen them violated? What problems arose from those violations?

10. Read this article: The Technical Skills You Need to Have as a Software Developer.

11. Read the book *Object-Oriented Programming in C# Succinctly* from Syncfusion.

12. ASSIGNMENT: Write code examples demonstrating each of the three pillars of object-oriented programming in C#.

13. ESSAY: Write an essay on the three pillars of object-oriented programming and how they reflect the real world.

14. Read this article: How to Solve Programming Problems.

15. Read the book *Regular Expressions Succinctly* from Syncfusion.

16. ASSIGNMENT: Write a console app in C# that uses regular expressions to find and replace instances of “double spaces” inside sentences. Then find an example of poorly written fiction on the Web and run the program against the fiction example text. Your program should list out the results of the work done - i.e., how many instances were found.

SECTION THREE
ADVANCED PROGRAMMING SKILLS

1. Read the book *Code Simplicity: The Fundamentals of Software* by Max Kanat-Alexander.

2. Read this article: The Role of a Senior Developer

3. Read this article: The Joel Test.

4. Read the book *Visual Studio 2017 Succinctly* from Syncfusion.

5. Read the book *HTTP Succinctly* from Syncfusion.

6. ASSIGNMENT: Find a friend or associate (ideally someone who doesn’t understand computers wells) and using a whiteboard, explain HTTP. Get feedback from them and write up what happened. Then do it a second time, with a different person, applying what you learned from the first experience. Write up what happened.

7. Read the book *ASP.NET Multitenant Applications Succinctly* from Syncfusion.

8. ASSIGNMENT: Complete this entire .NET-centric video course. Do all the coding in the course:
   a. Become a Full-stack .NET Developer
   b. Become a Full-stack .NET Developer – Advanced Topics
   c. Become a Full-stack .NET Developer – Architecture and Testing

9. Watch this video in full: Website Performance.

10. ASSIGNMENT: Using the data in that video, analyze the application you created just prior and improve its performance in some way. Write up what you did and why.

11. Read this article: Middleware Design Pattern

12. ASSIGNMENT: Write an article explaining what parts of the application you created in the previous .NET assignment could be considered “middleware”. Include a description of additional middleware elements you might add to the application, along with your reasoning for adding them in.

13. Watch this video in full: Microsoft Azure for .NET Developers – Cloud Patterns and Architecture.

14. Watch these videos in full:
    a. Providing Secure Cloud Applications
    b. Foundation for Cloud Architecture
    c. Cloud-native Architecture: The Big Picture

15. ASSIGNMENT: Complete the Getting Started with ASP.NET 4.5 Web Forms and Visual Studio 2013 in full.

16. ASSIGNMENT: Create the exact same application that you made in the previous step, using ASP.NET MVC, but NOT hosted on Azure. Note: Use Code First for the Entity Framework aspect.

17. ASSIGNMENT: Migrate that MVC application to Azure, including the database.

18. ASSIGNMENT: Make a change to the database schema. Migrate that change to the Azure database without losing any data in the Azure database.

19. Read the book *Localization for .NET Succinctly* from Syncfusion.

20. ASSIGNMENT: Add localization support for Spanish (Latin American) to the application you just created.

21. ESSAY: Write up what you did for the localization assignment above, and explain the benefits and drawbacks of the work.

22. Read the book *.NET Core Succinctly* from Syncfusion.

23. Read the book *ASP.NET Core Succinctly* from Syncfusion.

24. ASSIGNMENT: Rewrite the application you just made, using .NET Core.

25. Read the book *SQL Series Succinctly* from Syncfusion.

26. Read the book *SQL Server for C# Developers Succinctly* from Syncfusion.

27. ASSIGNMENT: Download the “Northwind” sample database from Microsoft.

28. ASSIGNMENT: Using the Northwind database, review the two previous books and write code against the database that uses each key concept from the books.

29. Watch this video in full: Security Architecture and Design: The Big Picture.

30. Watch this video in full: UX Fundamentals.

31. ESSAY: How could you change the UX for the application you deployed to Azure earlier to improve it?

32. Read the book *Unit Testing Succinctly* from Syncfusion.

33. Read this article in full: Beginner’s Guide to Unit Testing.

34. Read this article in full: Fundamentals of Unit Testing: Getting Started with Unit Testing.

35. Read this article in full, doing the coding as indicated: Fundamentals of Unit Testing: Test Your Application by Visual Studio Unit Test.

36. Read this article in full, doing the coding as indicated: ASP.NET MVC – Unit Testing.

37. ASSIGNMENT: Add a test project to one of your MVC applications from earlier in the course and add 10 unit tests to it.

38. Read this article in full, doing the coding as indicated: The Ultimate Guide To Unit Testing in ASP.NET MVC.

39. ASSIGNMENT: Refactor those unit tests from above to use RhinoMocks instead of Moq.

40. Read this article in full: Deploy an ASP.NET app to an Azure virtual machine.

41. ASSIGNMENT: Get a Microsoft Azure subscription (free ones exist). Using Azure, create a VM with the Windows Server 2016 operating system on it.

42. Watch this video in full: IIS Administration Fundamentals.

43. ASSIGNMENT: On your Windows Server VM, activate Internet Information Services if you haven't already.

44. ASSIGNMENT: Deploy a previously-made MVC application to that VM. Connect it to a SQL server database - either on that VM, or on Azure.

45. ASSIGNMENT: Make that web application accessible to remote users.

46. Read this article in full: Net Build Configuration System.

47. Read this article in full: Continuous integration, delivery and deployment in .NET projects.

48. ASSIGNMENT: Set up two environments using your Azure subscription. This should be two Windows Server 2016 machines that represent a "Development Environment" and a "Quality Assurance Environment". Each should have an associated SQL Server instance - NOT the same instance for each.

49. ASSIGNMENT: Using Jenkins, set up a CI system that automatically runs unit tests on Development Environment when you check in code to a repository.

50. ASSIGNMENT: Put one of your previous applications into this system you created.

51. ASSIGNMENT: Make sure the CI system works for successful commits, and for unsuccessful ones - intentionally submit a change that would break a unit test.

SECTION FOUR
QUALITY ASSURANCE

1. Read this article: Software Quality Assurance.

2. Read this article: Basic Terms and Concepts.

3. Read this article: Software Testing Levels.

4. Read this article: Software Testing Methods.

5. Read this article: Software Testing Types.

6. Read this article: Test Plan.

7. Read this article: Test Case.

8. Read this article: Test Script.

9. Read this article: Basics of Defects.

10. Read this article: Defect Severity.

11. Read this article: Defect Probability.

12. Read this article: Defect Priority.

13. Read this article: Defect Life Cycle.

14. Read this article: Defect Report.

15. ASSIGNMENT: Complete these exercises: Software Testing Exercises.

16. Read this article: Quality Assurance Best Practices.

17. Read chapters 2, 4, 5 and 6 of the book *Software QA Testing Career Package*.

SECTION FIVE
SOFTWARE ARCHITECTURE

1. Read this article in full: Software architecture.

2. ESSAY: How does software architecture relate to the daily actions of a software development team?

3. Complete this course in full: Developer to Architect.

4. ESSAY: What is your own purpose for learning about software architecture?

5. Read this article: Client/Server Computing.

6. ESSAY: Why was the client/server model created?

7. ASSIGNMENT: Sketch a typical client/server system, explaining how it works.

8. Complete this course in full: Understanding Enterprise Architecture.

9. ESSAY: Create a hypothetical situation where a piece of software is needed. Describe how you would apply the data in this video to the design of the software.

10. Complete this course in full: The Elements of Distributed Architecture.

11. Complete this course in full: .NET Distributed Systems Architecture.

12. Complete this course in full: Architecting Applications for the Real World in .NET.

13. ESSAY: Describe an application you have made. What are two alternate design patterns that could have been used to create that application? What are the advantages and disadvantages of each?

14. Complete this course in full: Domain-Driven Design.

15. ESSAY: Explain how you could apply the data in this video to a situation in which you were a consultant to a client who needed custom software.

16. Complete this course in full: Building End-to-End Multi-Client Service Oriented Applications.

17. ESSAY: What are the advantages and disadvantages of Service Oriented Architecture?

18. ASSIGNMENT: Research how SOA is implemented in three other languages. Write an essay explaining the differences and similarities between them, including the .NET approach you worked on earlier.

19. Complete these courses in full:
    a. Microservices: The Big Picture.
    b. Microservices Architecture.
    c. Microservices Architectural Design Patterns Playbook.

20. ASSIGNMENT: Create a small application using microservices. Do not make it complex; just create a working application.

21. Complete these courses in full:
    a. Azure Functions Fundamentals.
    b. Microsoft Azure Developer: Create Serverless Functions.

22. ASSIGNMENT: Create a simple serverless application on Azure.

23. Complete this course in full: AWS Developer: Serverless Architecture and Monitoring.

24. ASSIGNMENT:Recreate the same serverless application you made on Azure, but on AWS.

25. Read the book *Docker Succinctly* from Syncfusion.

26. Complete this course in full: Getting Started with Docker on Windows.

27. ASSIGNMENT: Create a simple .NET application and containerize it using Docker. Send it to another person and verify it can be installed and operated as expected.

SECTION SIX
SOFT SKILLS

1. Read *The Complete Software Developer's Career Guide* book by John Sonmez.

2. Read *Soft Skills: The Software Developer's Life Manual* by John Sonmez.

3. Complete this course in full: Picturing Architecture: UML (The Good Bits) and More.

4. ASSIGNMENT: Create a visual document, using the data in this video, that describes the architecture and operation of a standard ASP.NET MVC web application.

5. Complete this course in full: The Art and Practice of Information Architecture.

6. ESSAY: Describe three different types of information you could apply what you learned in this video to.

7. ASSIGNMENT: Pick one of those three types of information and create a visual document to describe it, using the data in this video and the video on picturing architecture.

8. Read this article in full: Painless Functional Specifications – Part 1: Why Bother?

9. Read this article in full: Painless Functional Specifications – Part 2: What's a Spec?

10. Watch this course in full: Technical Writing: Documentation on Software Projects.

11. ASSIGNMENT: Using the information from the video, create technical documentation for an application you created earlier in the course.

12. Read the book *Public Speaking for Geeks Succinctly* from Syncfusion.

13. ASSIGNMENT: Deliver a technical talk in front of an audience of at least 10 people. This can be done at The Tech Academy as one of our Tech Talks.

SECTION SEVEN
PLACEMENT

1. Read this article: Resumes.

2. Watch this video: Mid-Senior Dev Resumes.

3. Read Erik Gross's resume.

4. ASSIGNMENT: Write your resume to include all of the training and experience gained from this course. Send your resume to the VP for Quality (danny@learncodinganywhere.com) for review.

5. FINAL ESSAY: What did you gain from doing this course? What positive feedback would you like to provide? How can you utilize what you learned on this course? Email your essay to the Co-Founders (jack@learncodinganywhere.com and erik@learncodinganywhere.com).

Jack C. Stanley and Erik D. Gross
Co-Founders of The Tech Academy