ANTIOCH UNIVERSITY LOS ANGELES

UNDERGRADUATE STUDIES DIVISION

Course Syllabus

Course title:	Immersive Full Stack Web Development
Course prefix and number:	CSC 3010
Course Pre-requisite:	CSC 1010 Introduction to Full Stack Web Development
Number of units:	24
Quarter and Year:	Fall 2017
Day/Time:	40 hours a week of instructor led time Monday to Friday, 10 a.m. to 6 p.m.
Faculty name and degree:	Gregorio Rojas, BS Lenon Lopez, BA
Faculty availability (office hours):	
Faculty contact information:	To be provided by the instructor
Teaching Associate(s):	

COURSE DESCRIPTION:

The course will equip students to utilize a myriad of software development tools used by industry professionals to develop, debug and design dynamic web applications; conducting both front-end and back-end development, application program interfaces (API's), back-end data stores such as relational database management systems (RDMS) and team development strategies.

The course will prepare students to perform the duties of a typical entry-level Full Stack Web Developer. A Full Stack Web Developer is a programmer with a technical skill set that allows them to perform in client side, server side/middle tier, and backend-database development environments. Their main tasks are to develop, design and debug software that runs in a cross-browser environment served out of a web server backed by a database server for data persistence. After completing the course, the student will be substantially knowledgeable in the Software Development Life Cycle (SDLC) from concept to finished product and the ability to specialize, if they choose, in anything from front-end to back-end development technologies.

The course will teach students to be proficient in working in a full stack development environment. They will obtain the knowledge and skills to develop relational databases and work with data that is not stored in a relational manner. They will interact with their own API's and third party API's. They will also

be exposed to many User Interface and User Experience (UI and UX) () design concerns while building an understanding of how to gather customer and client requirements. They will be fluent in performing quality assurance testing and fundamental security concerns.

Specifically, the course will produce a programmer/developer that can work with many of the most popular design patterns, third party APIs, libraries and technologies including but not limited to: HTML5, CSS3, JavaScript, Bootstrap, MVVM, MVC, AngularJS, jQuery, Ajax, Inversion of Control, Dependency Injection, Principles of Object Oriented Design, Data Structures, Data Access, Database Design and Architecture, GIS, SMS, SMTP and RESTFul Api Design.

As part of the evaluation process of the student, the course will focus on a single, group based project that exposes the student to the full Software Development Life Cycle as experienced in a team practicing Agile/Scrum project management and development principles.

AULA UNDERGRADUATE PROGRAM LEARNING OBJECTIVES

All of AULA's undergraduate programs infuse curriculum with this purpose and these values through learning activities that cultivate the following intellectual and practical skills, applied learning, social awareness and responsibility:

- Critical and analytical thinking ability
- Ability to understand issues form multiple perspectives
- Ability to connect learning to lived experience
- Social and intercultural awareness
- Civic and community engagement
- Core competency in foundational skills

Web Development Program Learning Objectives

The AULA/Sabio Web Development program consists of a period of twenty-four (24) weeks during which students complete two 12 week intensive courses: "Intro to Full Stack Web Development" and "Immersive Full Stack Web Development". The overall learning objectives of the Web Development program are to develop competent Full Stack Web Developer professionals. After completing this program students will possess:

- Progressive beginning, intermediate, and advanced knowledge, skills, and abilities to lead the development and integration of state-of-the-art web software and systems.
- Abilities to function on all tiers of full stack web development (client/server/backend-database) in order to create and manage fully functional web based applications.

- Knowledge to apply the Software Development Life Cycle principles to design, develop, implement and maintain functional full stack web applications.
- Understanding of the changes that systems go through due to social and technological phenomenon, and how to cope with these factors.

COURSE LEARNING OBJECTIVES

After completing this intensive 12 week course titled “Immersive Full Stack Web Development”, our students will possess the knowledge, ability, and skills to develop and integrate software components for the development of web applications, performing as Full Stack Web Developers.

Web based application will be covered in depth leaving the student with the ability to create, modify and evaluate new and existing functionalities of web based applications.

The main learning objectives/outcomes of students completing the immersive course are:

- Ability, knowledge and skills to develop Web applications and projects, mastering the tools and software development skills needed to excel in all of the three major Web application tiers: Client side, Server Side (middle tier) and Backend (Database) development.
- Ability and skills to develop cross-browser applications targeting both desktop and mobile clients via responsive design through HTML5, CCS3, JavaScript, jQuery, Knockout.js or AngularJs or another client side MVC and/or MVVM application framework
- Proficiency in the effective use of web development applications and software tools such as: ASP.net C#/.Net or similar middle tier language and framework, RESTFUL API Design, MVC, Data Access, Fundamental Principles of Object-Oriented Design, Data Structures, Data Access, Singleton, Caching Strategies, Fundamentals of Database Design and Architecture, Key Table design considerations, Query Structure and Optimization, and Stored Procedures
- Ability and skills to build out a service oriented (SOA) application program interface (API) and the corresponding back end database.
- Ability to work with and integrate current leading developer tools and API's such as AWS, SendGrid, Google API's, Git, Trello, and Twilio, Browser Developer Tools and integrated developer environments (IDE’s) such as Visual Studio.
- Use of team communication and development strategies such as Agile/Scrum Methodologies used in Technical Project Management, Product Development, Team & Project Based Learning, Pair Programing, and Developer Code Reviews

Estimated time across the different languages during the course:

- HTML – 15%
- CSS – 10%
- JavaScript – 35%
- C# - 20%

- T-SQL – 20%

EVALUATION CRITERIA

Students should be able to perform the duties of a typical Full Stack Web Developer. A full stack web developer is a programmer with a technical skill set that allows them to perform in client side, server side/middle tier, and backend-database development environments.

Evaluation Criteria is based on the students proven abilities to develop, design and debug software that runs in a cross-browser environment on a web server backed by a database server:

- Faculty will evaluate the student's ability to effectively participate in all the steps from concept to finished product, and the ability to specialize in everything from front-end to back-end development through the class project and participation in solutions as part of a team, or as an individual.
- Students should demonstrate proficiency of the class work and lectures working in a full stack development environments.
- Students need to demonstrate the proven ability, knowledge, and skills through classwork and projects to develop relational databases and interact with third party API's outside their own application. They also need to be fluent in quality assurance, security concerns, and understanding customer and client needs through class participation and completion of all assignments, practice work, and lectures on these topics.
- Work done in class, class participation in key topics, and assigned projects, will be the main criteria used to evaluate students' knowledge of the multi-tier web environment and their implementation of software design patterns: APIs, libraries and technologies including but not limited to: HTML5, CSS3, JavaScript, Bootstrap, MVVM, MVC, Angularjs, jQuery, Ajax, Inversion of Control, Principles of Object Oriented Design, Data Structures, Data Access, Database Design and Architecture, GIS, SMS, SMTP and RESTFul Api Design.
- Students will participate in a mock interviews and other interview preparation where they will be presented with a variety of questions covering many of the topics covered in the course. Questions will require students to white board coding solutions, discuss open end software design questions as well as object oriented design questions regarding software development principles and practices. Students are expected to perform at a level that that would communicate their mastery of the topics.
- Students' past cards/tasks will continuously be evaluated throughout the course as we look for a progressive improvement in style, technique and application of the subject matters.
- There will be a Final Project due at the end of class; specifics of this to be shared and distributed in class.

ASSIGNMENTS

Assignments in this course follow agile software development practice so no one person is assigned any specific work as a general rule. The way it works, at a high level, is that we have a "list" or "backlog" of

tasks. As students finish their current task, they can pick up the next task on this list. If we see that any student needs to be exposed to more of a particular subject, then we break this protocol and help that student to learn the skills and do the work they need to do.

READING ASSIGNMENTS

Reading assignments can be found as part of the readings and projects the student will access through the wiki link:

https://sabiola.atlassian.net/wiki/pages/viewpage.action?spaceKey=TTS&title=Lists+by+Labels

Students can also enter this link for access to Sabio wiki dashboard:

https://sabiola.atlassian.net/wiki/dashboard.action

For Each Course Phase

- Reading Assignments will include at least 2-3 articles to be assigned, handed out in class, or posted as links.
- Writing/Project Assignments will include from 2-3 assignments/projects related to the week's topic and will be due with the specific expectations handed out in class.
- There will also be a Final Project due at the end of class; specifics of this will be shared and distributed in class.

As a first assignment students should follow these links on the Sabio wiki link to get acquainted with the material and where it is located. Students can gradually continue reading the support documents related to all phases of the Class Schedule in the order instructed in class.

- Welcome Package
- Find Sabio Tags in Starter Project
- How-To Articles
- Tools Home

CLASS SCHEDULE

This class schedule is for an intensive 12 week immersive program in Full Stack Web Development. Classes will meet for twelve weeks beginning, Monday-Friday 9 a.m. - 5 p.m. (40 hours with instructor per week for a class total of 480 hours). In addition, students are expected to spend 20-30 hours of independent work each week (leading to 60-70 total hours per student, per week for a total of ~780 hours for the class).

This section includes a general outline of each phase.

FIRST PHASE (WEEKS 1-2)

During this period the instructor will teach students to:

- Navigate through a solution comprised of multiple libraries internal and external
- Obtain familiarity with the principle of Agile Software Development
- Work in a team practicing Agile/Scrum project management

- Navigate through the Code of a complex HTML page
- Recognize, modify and enhance many of the major HTML Elements and CSS Rules
- Write JavaScript Functions that will affect the HTML page's element or their CSS rules
- Establish a connection to a database from an application
- Execute stored procedures within your application that accept and return parameters or datasets
- Implement a server side MVC framework
- Create and alter Stored Procedures
- Intro to Server Side Validation
- Intro to Client Side Validation
- Create T-SQL scripts to cover your typical CRUD operations
- Create and call Stored Procedures that either return parameters or dataset(s)
- Have a strong familiarity with the Management tools needed to perform all the tasks above.

Assignments will reflect achievement of the topics described in this phase.

Reading Assignments:

1. Read all information on JSON and other related documents in the Sabio wiki link that will be announced in class:

    a. JSON 101
    b. Validate JSON
    c. Ajax 101

2. Read all information on important C# topics link and other related documents that will be announced in class, Student should read the referred Links on C#:

    a. Object Oriented Basics C#
    b. Nullable types
    c.

3. Other required readings assigned by the instructor:

    a. Beginning MVC
    b. JavaScript Home
    c. Angular Home
    d. Angular Intro Guidance
    e. AngularJS
    f. C# Home
    g. C# Basics Quick Reference
    h. Sql Home
    i. Client Side Validation 101
    j. Server Side Validation 101

Writing/Project Assignments:
Complete the required tasks and practice exercises in the links

1. https://docs.angularjs.org/tutorial/

2. Checking In Code
3. Echoing Data: Your First Ajax Call
4. How to Use DataProvider - ExecuteNonQuery
5. https://sabiola.atlassian.net/wiki/display/TTS/My+First+Angular+Page

SECOND PHASE (WEEKS 3-6)

During this period the instructor will teach students to:

- Incorporate HTML, CSS, JavaScript, C#, T-SQL and form other sources into their own work
- Acquire an intermediate knowledge of the developer tools used to debug code on the server and the client
- Work in an environment under source control
- Interpret and implement third party libraries
- Integrate external API's into their own work via client integrations
- Implement a client side MV-VM or MVC framework
- Implement a client side library or framework that supports two-way binding
- Implement an Inversion of Control and Dependency Injection pattern on the client
- Integrate external API's into their own work via server side integrations
- Implement a web based Authenticate and Authorization
- Implement an Inversion of Control and Dependency Injection pattern in server side code

Assignments will reflect achievement of the topics described in this phase.

Reading Assignments:

1. MVC Beginner
    a. https://sabiola.atlassian.net/wiki/display/TTS/MVC+-+Razor+-+CSharp+Begginers
    b. MVC - Razor - CSharp Beginners

Read these links to get started with ASP.net's MVC framework:

1. Introduction to MVC Architecture and Separation of Concerns - read only this first link not "Part 2" or 3 etc.
    a. http://csharppulse.blogspot.in/2013/08/learning-mvc-part-1-introduction-to-mvc.html
2. MVC Overview (read only this first page)
    a. http://msdn.microsoft.com/en-us/library/dd381412(v=vs.108).aspx
3. Introduction to ASP.NET Web Programming Using the Razor Syntax (C#)
    a. http://www.asp.net/web-pages/tutorials/basics/2-introduction-to-asp-net-web-programming-using-the-razor-syntax
4. Action Results
    a. http://rachelappel.com/asp.net-mvc-actionresults-explained
5. MVC for Noobs
    a. http://code.tutsplus.com/tutorials/mvc-for-noobs--net-10488
6. Visual Studio Intro
    a. http://code.tutsplus.com/articles/visual-studio-web-dev-bliss--net-29477
7. Quick Syntax Reference (to be used as reference material)

a. https://docs.google.com/file/d/0ByDErZP3_PmOeHRBeFAzTG9OSFE/edit

Other Required Related Articles:

1. Simple Server Side Paging
2. Find Sabio Tags in Starter Project
3. C# Basics Quick Reference
4. Collections
5. Sending Email - Sendgrid
6. REST Api 101
7. Finally - Code Block
8. Exception/Error Handling
9. Mapping Sql and .Net DataTypes
10. Access Modifiers
11. Beginning MVC4 E-Book: Students will use the "MVC 4 Notes" document to help read through the MVC pdf book.
    a. Beginning MVC 4 Notes
    b. BeginningMVC4 01.pdf

Writing/Project Assignments
Assignments will reflect achievement of the topics described in this phase.

Complete the following tutorials:

1. MVC Noobs: http://code.tutsplus.com/tutorials/mvc-for-noobs--net-10488
2. Visual Studio Web Dev Bliss: http://code.tutsplus.com/articles/visual-studio-web-dev-bliss--net-29477
3. Practice Exception Handling:
    a. Exception/Error Handling

THIRD PHASE (WEEKS 7-9)
During this period the professor/instructor will teach students to:

- Debug an application across all tiers of development
- Create and alter Tables of a RDMS
- Create Foreign/Primary Keys on tables and discuss their use
- Nest stored procedure calls

Assignments will reflect achievement of the topics described in this phase.

Reading Assignments:
Client Side Validation

- Validation performed on the client side (browser) and the server side (C# tier).
    - https://sabiola.atlassian.net/wiki/display/TTS/Client+Side+Validation+101
    - This will take student to more related links.

Model Binding

1. Model Binding reading: Students should spend some time and read through the model binding links like the ones provided here. Knowing how to bind to complex objects beyond simple types and arrays is important.
    a. https://sabiola.atlassian.net/wiki/label/TTS/modelbinding

2. CSharp Quick Syntax:This is pretty decent collection of C# syntax. Students may prefer to just skim/read through this until they get more comfortable reading through the online documentation: CSharpQuickSyntax 01.pdf

Attribute Routing

1. https://sabiola.atlassian.net/wiki/display/TTS/Routing+By+Attributes+-+AttributeRouting
    a. Client-Side Routing

Rest Client Review:

Get the tools here for Chrome:
https://chrome.google.com/webstore/detail/advanced-rest-client/hgmloofddffdnphfgcellkdfbfbjeloo/details

To get to the client click on the "Apps" icon in chrome. It will be located in the top left of the screen.

Writing/Project Assignments:
Install the Chrome extension, then stop and read the wiki page which gives a detailed introduction on how to use it and the tasks that the student must complete:

https://sabiola.atlassian.net/wiki/display/TTS/Using+a+REST+Client.

1. Getting Started:
    a. http://screencast.com/t/pzz7GiOSLN -- this takes to this screen and click on this icon http://screencast.com/t/mjYYbPVIQCZi)
    b. http://screencast.com/t/l3sXSoJnt5

2. Routing: Students will be working to get a request routed to their controller. They will be working with routing. They will use the readings about Routing in WebApi or look through wiki for Routing.

FOURTH PHASE (WEEKS 10-12)
During this period the professor/instructor will teach students to:

- Discuss the benefits of Unit Testing and its place in the SDLC
- Implement design patterns into every day code writing
- Discuss the fundamental principles of object oriented programing
- Implementation and discussion of algorithms utilized in delivering high scale and available platforms

- Implement:
    - An application that implements session state
    - A web based application that maintains application state across sessions
    - Singleton Patterns
    - Server side caching
- Create new databases
- Create indexes and discuss their use

Assignments will reflect achievement of the topics described in this phase.

Reading Assignments:

Server Side Validation

1. https://sabiola.atlassian.net/wiki/display/TTS/Server+Side+Validation+101
    a. Overview Document: https://docs.google.com/document/d/1FqIET2RsIAedfCl04LnMC5D-B8I4ILkoWhcQ8Y0WT0w/edit?usp=sharing

Required Related Articles:

- Simple Server Side Paging
- AngularJS
- Data Validation
- Singleton in C#
- Value Types and Reference Types 2
- Geocoding
- Geographic Search
- Client-Side Routing
- Cache
- The Request Life Cycle
- Using View Models
- How to Call a Database
- Strings - Concat with Care

SQL Reading

- Students can get started on this this as soon as they have done both client and server side validation tasks.
    - Sql Home
    - https://sabiola.atlassian.net/wiki/display/TTS/SQL+Getting+Started
    - https://sabiola.atlassian.net/wiki/display/TTS/Mapping+Sql+and+.Net+DataTypes
    - Advanced SQL

Writing/Project Assignments:

1. Questions and Answers: Introduction to OOP and More: Please use following link:
    a. http://www.codeproject.com/Articles/22769/Introduction-to-Object-Oriented-Programming-Concep#Encapsulation%20
2. Experiment with Geographic Search at :Geographic Search
3. Students will be working on a Final Project due at the end of class; specifics of this to be shared and distributed in class.

REQUIRED READINGS

There is a "companion" wiki that Fellows can access for information, resources and custom tutorials. Readings can accessed through the wiki link:

https://sabiola.atlassian.net/wiki/pages/viewpage.action?spaceKey=TTS&title=Lists+by+Labels

Students can also enter this link for access to Sabio wiki dashboard:

https://sabiola.atlassian.net/wiki/dashboard.action

In addition there are a couple of books TBD.

Readings by topics:

Agile Readings
Agile Scrum Primer

- https://sabiola.atlassian.net/wiki/display/TTS/Agile+Scrum+Primer

More about Agile/Scrum

- http://agilemanifesto.org/
- http://agilemanifesto.org/principles.html
- https://docs.google.com/document/d/1jDn-FdNhUEk1pUuZlbrN3UYP2iKbYiBgXk6Lf-0hgmQ/edit

JSON and Ajax Readings
Links under the JSON label in our wiki:

- JSON 101
    - https://sabiola.atlassian.net/wiki/display/TTS/JSON+101
- Validating a JSON object
    - https://sabiola.atlassian.net/wiki/display/TTS/Validate+JSON
    - The above section of the wiki contains information about client side validation, server side validation, then making the student's first AJAX call.
- Ajax 101
    - https://sabiola.atlassian.net/wiki/display/TTS/Ajax+101

Important C# Topics
Links on C#

- Object-oriented basics:
  https://sabiola.atlassian.net/wiki/pages/viewpage.action?pageId=5308419
    - C# Types
    - Nullable types:

  https://sabiola.atlassian.net/wiki/display/TTS/Mapping+Sql+and+.Net+DataTypes

MVC Beginner

https://sabiola.atlassian.net/wiki/display/TTS/MVC+-+Razor+-+CSharp+Begginers

MVC - Razor - CSharp Beginners

Links to get started with ASP.net's MVC framework.

- Introduction to MVC Architecture and Separation of Concerns - read only this first link not "Part 2" or 3 etc.
  http://csharppulse.blogspot.in/2013/08/learning-mvc-part-1-introduction-to-mvc.html
- MVC Overview (read only this first page)
  http://msdn.microsoft.com/en-us/library/dd381412(v=vs.108).aspx
- Introduction to ASP.NET Web Programming Using the Razor Syntax (C#)
  http://www.asp.net/web-pages/tutorials/basics/2-introduction-to-asp-net-web-programming-using-the-razor-syntax
- Action Results
  http://rachelappel.com/asp.net-mvc-actionresults-explained
- MVC for Noobs
  http://code.tutsplus.com/tutorials/mvc-for-noobs--net-10488
- Visual Studio Intro
  http://code.tutsplus.com/articles/visual-studio-web-dev-bliss--net-29477
- Quick Syntax Reference (to be used as reference material)
  https://docs.google.com/file/d/0ByDErZP3_PmOeHRBeFAzTG9OSFE/edit

Required Related articles

- Simple Server Side Paging
- Find Sabio Tags in Starter Project
- Singleton in C#
- Value Types and Reference Types 2
- C# Basics Quick Reference

Beginning MVC4 E-Book

Students will use the "MVC 4 Notes" document to help read through the MVC pdf book.

BEGINNING MVC 4 NOTES

BEGINNINGMVC4 01.PDF

Client Side Validation

Validation performed on the client side (browser) and the server side (C# tier).

- https://sabiola.atlassian.net/wiki/display/TTS/Client+Side+Validation+101

This will take student to more related links.

Model Binding

Model Binding reading: Students should spend some time and read through the model binding links like the ones provided here. Knowing how to bind to complex objects beyond simple types and arrays is important.

- https://sabiola.atlassian.net/wiki/label/TTS/modelbinding

CSharp Quick Syntax:

This is pretty decent collection of C# syntax. Students may prefer to just skim/read through this until they get more comfortable reading through the online documentation.

- CSharpQuickSyntax 01.pdf

Attribute Routing

- https://sabiola.atlassian.net/wiki/display/TTS/Routing+By+Attributes+-+AttributeRouting

Rest Client Review

- Get the tools here for Chrome

https://chrome.google.com/webstore/detail/advanced-rest-client/hgmloofddffdnphfgcellkdfbfbjeloo/details

- To get to the client click on the "Apps" icon in chrome. It will be located in the top left of the screen.
- Once the student has installed the Chrome extension they should stop and read the wiki page which gives a detailed introduction on how to use it:

https://sabiola.atlassian.net/wiki/display/TTS/Using+a+REST+Client

- Getting Started
    - http://screencast.com/t/pzz7GiOSLN This takes to this screen and click on this icon http://screencast.com/t/mjYYbPVIQCZi)
    - http://screencast.com/t/l3sXSoJnt5

Server Side Validation

https://sabiola.atlassian.net/wiki/display/TTS/Server+Side+Validation+101

Overview Document

- https://docs.google.com/document/d/1FqIET2RsIAedfCl04LnMC5D-B8I4ILkoWhcQ8Y0WT0w/edit?usp=sharing

Related articles

- Simple Server Side Paging

- AngularJS
- Data Validation
- Singleton in C#
- Value Types and Reference Types 2

SQL Reading

Students can get started on this this as soon as they have done both client and server side validation tasks, or save this for later.

- https://sabiola.atlassian.net/wiki/display/TTS/SQL+Getting+Started
- https://sabiola.atlassian.net/wiki/display/TTS/Mapping+Sql+and+.Net+DataTypes

Other Required Readings:

- Welcome Package
- Find Sabio Tags in Starter Project
- How-To Articles
- Tools Home
- Beginning MVC
- Cache
- Geocoding
- Client-Side Routing
- JavaScript Home
- Angular Home
- Angular Intro Guidance
- AngularJS
- C# Home
- Sql Home
- Data Validation
- What to do after Training

OTHER COURSE AND UNIVERSITY POLICIES

Attendance Policy

University policy states: "students are expected to attend all class sessions and, for online courses, participate in online discussions as required in the syllabus. If a student attends less than 80% of class sessions, the student will earn no credit for the course. If a student anticipates an absence for religious observance, work obligations, or any other reason, the student consults with the instructor before or during the first week of class to request an accommodation in the form of makeup assignments. In some cases, however, accommodation may not be possible if in the instructor's judgment the absence would be disruptive to the learning process. In these cases the judgment of the instructor is final." (See AULA General Catalog, http://aulacatalog.antioch.edu/policiesregulationsandprocedures/academicpolicies

You are expected to attend all class sessions. If an absence is unavoidable, please contact me about a make-up assignment. This course program requires a highly intensive learning environment with accumulation of knowledge that depends on each previous assignment. Thus, the attendance policy reflects this need, and you will not receive credit if you miss more than 3 days or 24 hours of instructor-led class time.

Letter Grade Equivalent Policy

If you want a grade equivalent, please request it in writing by week two.

Extra Credit Policy

Extra units are not allowed.

Incomplete Policy

Incompletes are not available for this course. If you are not able to complete your assignments as required in the syllabus and as stipulated in the course meetings, you may not be able to receive credit for this course.

Information Literacy and Research Requirements

All students are expected to develop an understanding of how to find and use resources appropriate for academic inquiry and scholarship. Although these are not skills that are required for the completion of this course, please be aware that students may meet with the AULA librarian should they need any help and support with research and for information literacy instruction.

Student Conduct Policy

Please provide expectations for student conduct in your class. EXAMPLE: "Respectful conduct is expected of students on the campus at all times, both inside and outside the classroom." See *AULA General Catalog,* http://aulacatalog.antioch.edu/policiesregulationsandprocedures/studentconduct/ for university policy.

Plagiarism Policy

University policy describes plagiarism as "the representation of someone else's writing, graphics, research, or ideas as one's own. Paraphrasing an author's ideas or quoting even limited portions of the work of others without proper citation are also plagiarism, as is cutting and pasting materials from the Internet into one's academic papers. Extreme forms of plagiarism include submitting a paper written by another person or

purchased from a commercial source." (See *AULA General Catalog,* ***http://aulacatalog.antioch.edu/policiesregulationsandprocedures/formsofprobationandtheirconsequences/***)

AULA does recognize that computer programming requires a specific interpretation of this policy to be consistent with accepted practices in the field. It is accepted that students will collaborate and share knowledge and code, including using code from other successful applications. However, it is expected that students make acknowledgements and, perhaps most importantly, be able to explain and defend the solutions created. If a student is unable to provide these types of learning evidence, this may be cause to suspect plagiarism and then the Antioch University policy regarding this may be followed. Additionally, we reference the "Guidance Notes on Plagiarism" from the Department of Computer Science at the University of Birmingham:

(http://www.cs.bham.ac.uk/internal/studentinfo/plagiarism.htm):

Avoiding Plagiarism in Computer Programs

Almost all computer programs contain many ideas borrowed from elsewhere. Many also contain short sections of actual code copied from elsewhere. For example, writing a section of program to create a new window on screen with a menu at the top of the window is often done by simply copying a few of lines of code from an example in a programming manual or textbook, either with or without a few minor changes. This is normally regarded as fair use and typically requires no acknowledgement.

Any more significant copying of code from elsewhere should be acknowledged, however. The acknowledgement can be put in comments within the program itself. Reference to the source of the original material should be made in the same way as in essays or other documents (except that it may not be possible to use italics or other font variations). Obviously, it is not possible to put sections of code in quotation marks to indicate that they have been taken directly from elsewhere. Instead, the comments should make it clear which sections of code have been copied from elsewhere. Equally, the comments should make it clear when the basic method has been copied from elsewhere, but changes made to the details.

Reasonable Accommodation for Students with Disabilities

Antioch University is committed to providing reasonable accommodations to qualified students with disabilities in accordance with Section 504 of the Rehabilitation Act of 1973 and the Americans with Disabilities Act of 2008. Students who need to request disability accommodations should email or call Yaru Wang, disabled student services coordinator (dss.aula@antioch.edu or 310-578-1080 x 209) at the outset of their enrollment, if possible, since reasonable accommodations are not retroactive.

Sexual Harassment Policy

The Undergraduate Studies Division is firmly committed to each student's dignity and to eliminating all forms of sex discrimination and harassment of students. No student should have her or his learning experience at AULA contaminated by the experience of being treated as a sexual object by an instructor or any other employee. We strongly urge any student who believes that an Antioch employee has crossed the line to speak to your advisor, to the Undergraduate Studies Division leadership, the Provost, or the Director of Human Resources about your concerns.

Antioch University's policy "Title IX, Sex Discrimination, Sexual Harassment, and Sexual Violence" provides definitions of prohibited and inappropriate behaviors, the process for reporting and investigating complaints, and the sanctions levied against those employees or students found to be in violation of these policies. This policy can be found in the Antioch University Resource Archive at http://aura.antioch.edu/policies_400_6x/12/.

Additionally, please visit the link below for Antioch University's policy on dual relationships:

http://aura.antioch.edu/policies_400_6x/11/

Antioch University Policies:

Antioch University is committed to building a vibrant and inclusive educational environment that promotes learning and the free exchange of ideas. Our academic and learning communities are based upon the expectation that their members uphold the shared goal of academic excellence through honesty, integrity, and pride in one's own academic efforts and respectful treatment of the academic efforts of others.

All students are expected to comply with Antioch University policies, including the Title IX Sexual Harassment and Sexual Violence Policy and the Student Conduct Policy.

To access academic, student, and other university policies are available online: http://aura.antioch.edu/au_policies/