Professional Self-Assessment

Anthony Baratti

Southern New Hampshire University

CS-499 Computer Science Capstone

Professor J. Lusby

August 17, 2025

Professional Self-Assessment

My name is Anthony Baratti, and I am a Software Engineering Student at Southern New Hampshire University (2022-2025).

I have studied Client-Server Development, Data Structures & Algorithms, Emerging Technologies (Artificial Intelligence), Full Stack Development (MongoDB, Express, AngularJS, and Node.js) & Cloud Deployment (Docker and AWS), Secure Coding Principles, Mobile Architecture & Application Development, System Analysis & UI/UX Design, Reverse Engineering, Testing, Computer Graphics (OpenGL), and Micro-Controller Programming (state machine).

I have used Java, Python, C/C++, JavaScript, TypeScript, and have used JSON and SQL modeling to create and manage databases in MongoDB, AWS, and SQLite3 with knowledge of tools such as Postman and MongoDB Compass.

I have used Eclipse, PyCharm, Jupyter Notebook, Spyder, Microsoft Visual Studio, Visual Studio Code, and Android Studio, among many other IDEs. I have developed cloud services using the AWS platform, using S3, Lambda Functions, and API Gateways.

During my academic career at Southern New Hampshire University, applying best practices was a critical foundation for success. Detailed commenting and documentation, thorough unit testing,

UI and UX design (including platform-specific mobile and wearable design), diagram design and completeness, developing a security mindset (such as defense by design, layered defense, and principle of least privilege, OWASP dependency checks, etc.), modularity (separation of concerns), portability (building re-usable components), scalability, determining best architectures, planning and pseudocode, and much more. Combining all of these skill sets has allowed me to practice not just productive coding but has honed those skill sets into well-rounded and versatile tools that can be applied to add value to software engineering and development. Reading and writing code became second nature as the ability to plan and construct robust systems with complete documentation and functionality paved the foundation for excellence.

Object-oriented programming (OOP) created an environment in which principles such as encapsulation, inheritance, abstraction, and polymorphism (such as method overloading) allowed for a creative mix of problem-solving solutions that helped construct dynamic and reusable code.

Discovering a plethora of libraries such as Python's NumPy, Pandas, Keras, and TensorFlow backend, Java's Spring Framework and JUnit testing framework, JavaScript with Node.js framework such as Express and React, and the C++ OpenGL library. Using a combination of programming languages and their appropriate libraries enabled me to explore the possibilities of multi-language and multi-library development.

I also undertook studies in the software development life cycle, where planning, development, testing, deployment, monitoring, and repeating revealed the mechanics of a continuous integration and continuous development (CI/CD) pipeline. From planning using user stories, diagrams (use case, process, component, data flow, sequence, activity, state machine, class, deployment, API, and call graph diagrams), pseudocode, business requirements documents (such as acceptance criteria, technical specifications, system requirements, etc.), wireframes, and many more tools. Comparisons between different development styles were also revealed, such as the Agile-Scrum methodology (utilizing short sprints for development) and the Waterfall methodology (which involves planning, coding, testing, and deploying), providing insight into when each methodology is best implemented.