



Attachment 1

Full Name: Ardine Martine Nukuri

Grading system ⁽¹⁾: Grades are awarded on a 5.0 point scale, corresponding to percentage scores. A Level 5 (over 85%) indicates mastery, Level 4 (65%-84.99%) indicates proficiency, and Level 3 (50%-64.99%) indicates application ability. The final grade is presented as a percentage.

DESCRIPTION OF ATTENDED COURSES ⁽²⁾

Course Unit Title ⁽³⁾	Main contents ⁽⁴⁾ [min. 100, max. 500 characters]	Books used ⁽⁵⁾	Hour S ⁽⁶⁾	Grad e ⁽³⁾
Learning Processes	Basic level course in metacognition and learning strategies. Tackled learning strategies attuned to critical thinking, breaking information down into manageable pieces, retention, and learning in high demand courses. Building a personal framework for adaptable learning was highlighted.	Make It Stick: The Science of Successful Learning by Brown, Roediger & McDaniel.	240	84.86%
Reflective Thinking	Self assessment and iterative learning were the focal points. Integration of learning new cognitive offloading biases, performance analysis, feedback for learning and development were used in the course for development for self and for others. Development concerned was personal and professional. Emphasis on reflection and journaling for development were noted.	University provided materials; Reflective Practice journal articles.	240	79.97%



Self Leadership and Team Dynamics	Cooperative principles course. Explored and practiced diverse leadership approaches, conflict management, and	The Five Dysfunctions of a Team by Patrick Lencioni.	160	79.67%
	agile methodologies. Aimed to develop the interpersonal components to pass and lead development in heterogenous software development teams.			
FCSE: Programming	How to write code with a more algorithmic mindset. How to use programming to solve different problems. Understanding programming as a language and mastery of different variables used in the programming language. Understanding programming constructs (loops, functions, conditionals, arrays, etc.) and the importance of problem solving with programming in a clean and articulate manner. Understanding the basic data structures and concepts of programming and algorithmic thinking.	Automate the Boring Stuff with Python; Online coding platforms (HackerRank).	320	87.96%
Responsible Enterprise	Developed an understanding of the ethics of technology, the business of technology, and the social responsibility of technology. Provided an overview of concepts of corporate social responsibility (CSR), ethics in decision making, and the impact of technology in society. Purpose of profit and social responsibility.	Business Ethics: A Managerial Approach by Wicks, Freeman, et al.; Industry case studies.	160	79.90%
FCSE: Webstack	Understanding of web technology. Understanding and use of client programming with HTML, CSS, JavaScript. Understanding of server programming, programming with SQL, and understanding the client-server model (HTTP).	MDN Web Docs; Web Development with Node and Express by Ethan Brown.	320	96.53%



Foundations Project	First of the many project-based courses. Understanding from the previous modules and applying them to real-world problems collaboratively. Each group used the Software Development Life Cycle (SDLC) to a real-world problem. The course divided the time to project evaluation and project management with collaboration.	The Mythical Man-Month by Frederick Brooks; Agile project tools	160	88.50%
Communicating for Impact	Courses on professional and technical communication skills. Instructed on public speaking and presentation skills, technical writing to diverse audiences, and data visual communication. Important to	Made to Stick by Chip & Dan Heath; Public speaking guides and TED Talks.	160	85.14%

	conveying professional and technical information for the first time.			
Enterprise Web Development	Developed web skills to build and manage large and complex applications. Advanced web architecture, user authentication and authorization systems, API security, and deployment to enterprise-level applications. Constructed and maintained web services with scalable.	Node.js Design Patterns; OpenAPI Specification guides.	160	78.16%
Skills Immersion 1	Practical course that imitates a working software development environment. Mastered tools and workflows for the present day, like Git and collaborators for version control, CI/CD, and agile/scrum.	Atlassian tutorials; The Pragmatic Programmer by David Thomas and Andrew Hunt.	240	92.43%
Introduction to Software Engineering	The principles and practices of engineering software of good quality. The full spectrum of the SDLC, and tracking the work done in the software with UML, various models like agile or waterfall, software testing, maintenance in the long run.	Software Engineering by Ian Sommerville; UML specification guides.	240	81.90%



Advanced Python Programming	Learning Python for software engineering in great detail. Included advanced-level data structures like object-oriented programming (OOP) and concepts from functional programming like decorators and algorithm-implementing generators. There was a focus on writing effective and scalable Pythonic code that was clean in applications that were complex.	Fluent Python by Luciano Ramalho; Python Cookbook by Beazley & Jones.	240	89.15%
Advanced Back-End Development	Focused on the design and construction of strong applications for the server-side. There were lessons on the design of RESTful APIs and other topics like database building, user authentication, system scalability, and microservices. There was practical application of the lessons using Node.js and Django.	Designing Data-Intensive Applications by M. Kleppmann; Microservices Patterns.	240	93.00%
Mobile Application Development	In-depth studies of mobile app development, both native and crossplatform. Included topics of user interface	Official Flutter & Dart Documentation; Flutter for	240	98.80%

	(UI) and user experience (UX) design, state management, and working with hardware for devices like cameras and GPRS, as well as consuming REST APIs. The end of the course featured the construction of a mobile application that was fully functional, from the initial idea to deployment.	Beginners by A. Biessek.		
Advanced Frontend Web Development	The study of advanced architecture on the front end, as well as the modern JavaScript framework. There was a focus on the building of interfaces that are useroriented and also interactive as well as on state management that was complex, for example, Redux. There were other topics tackled as well, like the optimization of performance for the front end, design tools, and testing.	You Don't Know JS series by Kyle Simpson; Eloquent JavaScript by M. Haverbeke.	240	93.90%



React Development	A very detailed course focused on one of the most popular libraries for developing single-page applications: React. We spent a lot of time learning the core pieces of React, including Component Structure, JSX Mark up, Props, and State; as well as learning about Hooks, the Context API, and Client Side Routing. As a result, I was able to create a number of user interfaces for different applications.	Official React Documentation; The Road to React by Robin Wieruch	240	95.75%
Advanced DevOps	Courses related to the principles and practices of DevOps to create a culture of automation for the Software Delivery Lifecycle. Related to Continuous Integration and Continuous Delivery (CI/CD) Automation, Infrastructure as Code (IaC) with tools like docker and kubernetes, and monitoring and Logging of Production systems.	Specialized Coursera courses; Docker & Kubernetes official documentation. Coursera courses	240	92.30%
Ethics in Software Engineering	Courses related to the professional and ethical challenges of the software development discipline. Related to Data Privacy (GDPR), Algorithmic Bias, Software and System Social Impacts, IP, and The Ethical Engineering Practice.	ACM Code of Ethics; Ethics for the Information Age by Michael J. Quinn.	240	84.82%
Mission Capstone	The final, year-long capstone project, requiring students to design, develop, and deploy a significant software solution for a real-world client or problem. This comprehensive project demonstrated mastery of the entire software engineering lifecycle, from ideation and project management to final delivery.	Project-specific research and technical documentation.	240	78.31%

I declare under my full responsibility that I have given correct and true information on all of the above.

Date and place

Signature