Contact

0842311730 (Mobile)
angeliquehilario019@gmail.com

www.linkedin.com/in/angelique-hilario-lique007 (LinkedIn)
github.com/DevLique (Portfolio)

Top Skills

Microsoft Office
High Level Of Accuracy
HTML5

Certifications

Intro to Docker
Generative AI with Large Language Models
AI For Everyone
Active Listening: Enhancing Communication Skills
Developing Interpersonal Skills

Angelique Hilario

Aspiring Software Developer | WeThinkCode _Alumni
City of Cape Town, Western Cape, South Africa

Summary

My passion for technology ignited in childhood, sparking a lifelong fascination with coding and innovation. Over the years, I've explored various career paths, each enriching my perspective and deepening my appreciation for the power of code. These diverse experiences have not only broadened my skill set but also reinforced my belief in technology's potential to drive impactful solutions. Today, I'm excited to leverage my broad background and coding expertise to build transformative projects and tackle new challenges.

Experience

CAPACITI
Digital Associate
July 2025 - Present (3 months)

JMA Enterprises
Administrative Assistant
June 2020 - November 2020 (6 months)
Western Cape, South Africa

assist with business related queries (CIPC , SAGE, WAVE, SARS, MS Office)
assist with bookkeeping for businesses.
workout quotes , attend to clients via email and face to face in a friendly and professional manner.

AVI Marketing
Merchandiser
March 2014 - December 2014 (10 months)
South Africa

eyesave optometrists
optometrists assitant
April 2013 - October 2013 (7 months)
South Africa

Education

WeThinkCode

NQF 5, Computer Science · (September 2023 - December 2024)

malibu high school

· (2008 - 2012)

132 Jan Smuts Avenue
Parkwood, Johannesburg
2193
www.wethinkcode.co.za

2 April 2025

To whom it may concern

This confirms that **Angelique Hilario** with ID/Asylum Seeker number **9412090185081** has successfully concluded the 16-month software engineering course with WeThinkCode_ as of December 2024. **Angelique** has fulfilled all the program requirements and demonstrated the necessary competencies.

The WeThinkCode_ accreditation details with **MICTSETA** are as follows:

ID	QUALIFICATIONS TITLE	NQF LEVEL	CREDITS
48872	National Certificate: Information Technology (Systems Development)	Level 5	131

Should you require any further information regarding our graduates or curriculum, please feel free to contact us.

Sincerely,

Kelebogile Motlhamme

Director of Operations

WeThinkCode_ Student Transcript

WeThinkCode_ (Holdings NPC) certify that **Angelique Hilario** was registered as detailed below:

Full Name: **Angelique Hilario**
ID/Passport: **9412090185081**
Student Username: **ahilario023**
Cohort: **2023**
Campus: **Cape Town**

Qualification Details

ID	48872
Qualifications Title	National Certificate: Information Technology (Systems Development)
NQF Level	Level 5
Credits	131

Year 1			
Semester 1	Fundamentals	**81**	
Semester 2	Object Oriented Programming	**64**	
Year 2		**Module Score**	**Semester Score**
Semester 3	Brownfields Development	**72**	**69**
	Web Development	**60**	
Semester 4	Elective: Mobile Development	**49**	**49**

Year 2 Final Score
60
Final Outcome
Pass

WeThinkCode_ (Holdings NPC) hereby certifies that these are the official results for Angelique Hilario with ID: 9412090185081 has concluded the 16-Month software engineering course with WeThinkCode_ with a final result stipulated above.

Performance Scale Descriptors

Performance Scale Descriptors					
Level	**5 (100%)**	**4 (80%)**	**3 (60%)**	**2 (40%)**	**1 (20%)**
Outcome	Exceeds Expectations	Above Expectations	Meets Expectations	Below Expectations	Expectations Not Met

Interpreting the Outcomes

Exceeds Expectations: Student is able to submit project requirements well in advance before the deadline. Grading was passed/correct on the first attempt.

Above Expectations: Student attempted grading twice before the deadline, but passed on the second attempt before the deadline.

Meets Expectations: Student is able to submit projects on time and pass at the required level.

Below Expectations: Student is below expectations, attempted grading and submitted after the deadline, and passed.

Expectations not Met: student attempted to submit after the deadline passed/failed and this indicates that major development is
required. Students should make use of the interventions available to improve performance.

Progress Report

Fundamentals		
Making Decisions	Students apply boolean expressions and conditional statements to control program flow and handle different conditions, focusing on clear and effective conditional logic.	5
Repeating Instructions	Students learn to use loops (while, for) to repeat code efficiently, handle lists, and control loop flow with break and continue. The project also covers simulating do..while loops in Python, focusing on combining loops with conditionals for versatile solutions.	5
Structuring Data	Students learn to use data types like tuples, lists, sets, and maps to organise and manipulate collections of values efficiently.	4
Combining Instructions	Students explored how functions can return values to enhance code flexibility and reusability. They practised defining and using functions for basic operations, delved into functional composition by nesting function calls, and introduced recursion to break down complex problems into simpler tasks. These techniques contribute to creating more organised and efficient code.	3.8
Processing Collections	Students explore advanced methods for processing data collections in Python. They learn to use lambda functions, along with map(), filter(), and reduce(), for efficient data handling. They also practise list and dictionary comprehensions to streamline code and enhance readability, offering cleaner alternatives to traditional loops.	3.5
Modules And Packages	Students learn to break down complex programs into manageable modules, improving simplicity and maintainability through modular programming. They gain an understanding of using namespaces to organise code and prevent name clashes. Additionally, they cover creating and utilising packages to structure modules effectively and manage dependencies.	4.3
Don't Panic (Errors And Exceptions)	Students learn to identify and handle syntax errors and runtime exceptions. They practise debugging techniques, use assertions to verify code assumptions and manage exceptions with try-except blocks. Additionally, students explore raising exceptions to signal unmanageable error conditions.	5

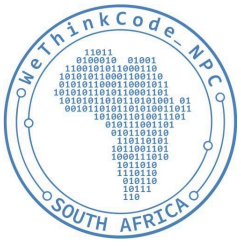
Code Clinics (Introduction To APIs)	Students are expected to work iteratively within a given timeline to develop a command-line program for automating a booking system. They will collaborate in teams, utilising tools such as Git for version control and Miro for project planning and tracking. Integration with Google's APIs is required, and the project will follow an agile approach to manage development and progress.	4.2
Object-Oriented Programming		
Encapsulation	Students learn to define and use classes and objects, ensuring modularity by hiding internal states and interacting through methods. Learning outcomes include creating classes, enforcing data hiding with private variables, using constructors, following Java naming conventions, and applying encapsulation principles for maintainable and scalable code.	4.1
Polymorphism, Composition And Inheritance	Students learn to integrate polymorphism, inheritance, and composition to build flexible and maintainable software. They apply inheritance for code reuse, use composition for creating complex objects, and leverage polymorphism to treat diverse objects uniformly, leading to more modular and adaptable code.	5
Robot Worlds (Greenfields Development)	Students engage in a greenfields project to develop a client-server Robot World system, working iteratively in agile sprints. This project applies OOP principles, including inheritance, composition, and polymorphism, to create a world where robots interact with obstacles and each other. The project involves implementing network programming, concurrency, and application protocols, providing hands-on experience with Java sockets and JSON serialisation.	4

Brownfields Development		
Acceptance Testing	Students test the functionality of their software to ensure it meets client, user, or stakeholder expectations. Acceptance testing, which combines integration and UI tests, is conducted using frameworks like JUnit.	4.6
Build Pipelines	Students integrate code changes and establish a Continuous Integration/Continuous Delivery (CI/CD) pipeline. This includes creating a build script that runs with new commits and extending it to package the server for deployment using Docker.	4.5
Relational Databases & Persistence	Students learn the fundamentals of relational databases, including tables, rows, columns, primary keys, and foreign keys. They gain proficiency in Structured Query Language (SQL) and explore its integration into Java applications through the Java Database Connectivity (JDBC) API.	3.9
Separation Of Concerns: Designing For Change	Students apply architectural patterns to enhance system design by improving modularity and maintainability. This involves evaluating and making design decisions based on architectural patterns and implementing a Web API for the RobotWorld server. They also explore Object-Relational Mapping (ORM) principles for better data access management.	2.4
Web Development		
WebDev Exercise	Students develop a client-side user interface that interacts with an API over HTTP, using HTML, CSS, and JavaScript. Proficiency in manipulating the Document Object Model (DOM) and handling data with XMLHttpRequest or the Fetch API is essential.	5
Client Side Rendering - SPA(Single Page Web Application)	Students develop a client-side user interface that interacts with an API over HTTP, using HTML, CSS, and JavaScript. Proficiency in manipulating the Document Object Model (DOM) and handling data with XMLHttpRequest or the Fetch API is essential.	2
Server-Side Rendering: WeShare MVC	Students work on a web-server backend that serves HTML, CSS, and JavaScript to the browser and processes user inputs through form-handling endpoints. They also explore template-generated HTML with Thymeleaf and apply the Model-View-Controller (MVC) architecture.	3
Elective		
Mobile Development	Students learn to design and develop mobile applications using Flutter, focusing on visual and architectural design. They explore	49

UI and UX design principles to create intuitive and engaging user interfaces and use asynchronous programming techniques and Firebase integration for data handling. Through state management, they maintain consistency across app components. Additionally, they build a Minimum Viable Product (MVP), applying knowledge in Flutter and asynchronous programming to deliver a functional app. This module emphasises practical application through an individual project and reinforces key concepts in mobile app architecture and state management.

Assessments		
Semester 1	**Python_1**	74
	Python_2	73
Semester 2	**Formative_1**	66
	Formative_2	50
	Summative_1	52
Semester 4	**Formative**	50
	Summative	56

I do hereby certify that these are the official results for as of 22/01/2025.



Corban Loots
Performance Lead, Johannesburg
corban@wethinkcode.co.za