## AGENDA

- Presentation of myself
- Structure of the course
- Software development skills to create AI applications
- Just a kick off session Python VS C++

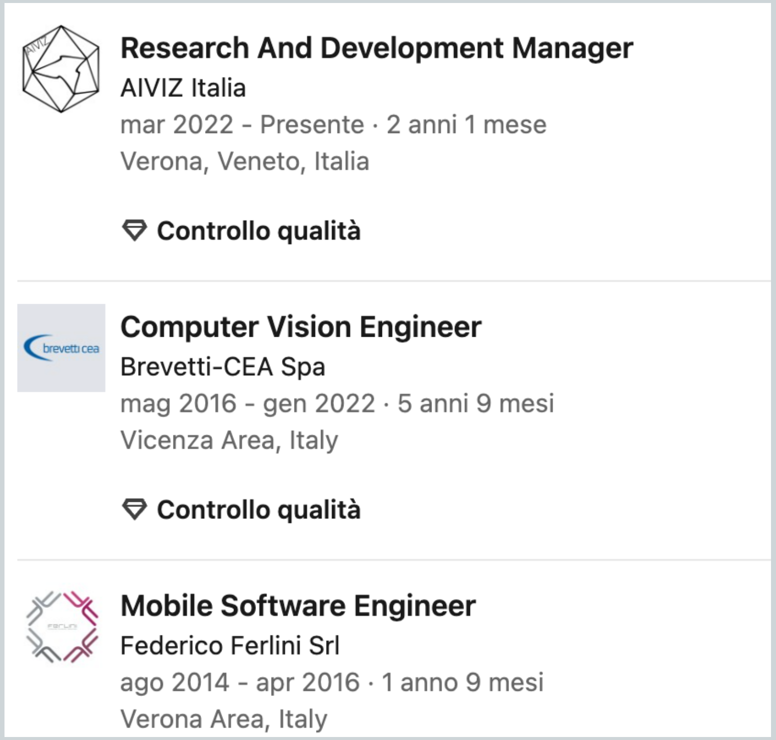
## Presentation of myself

Pasquale.Mirtuono@univr.it

Pasquale Mirtuono
Inspection Vision Expert, PhD
Verona, Veneto, Italia · Informazioni di contatto
http://www.mirtuono.it

AIVIZ Italia
Università degli Studi di Verona

## Presentation of myself



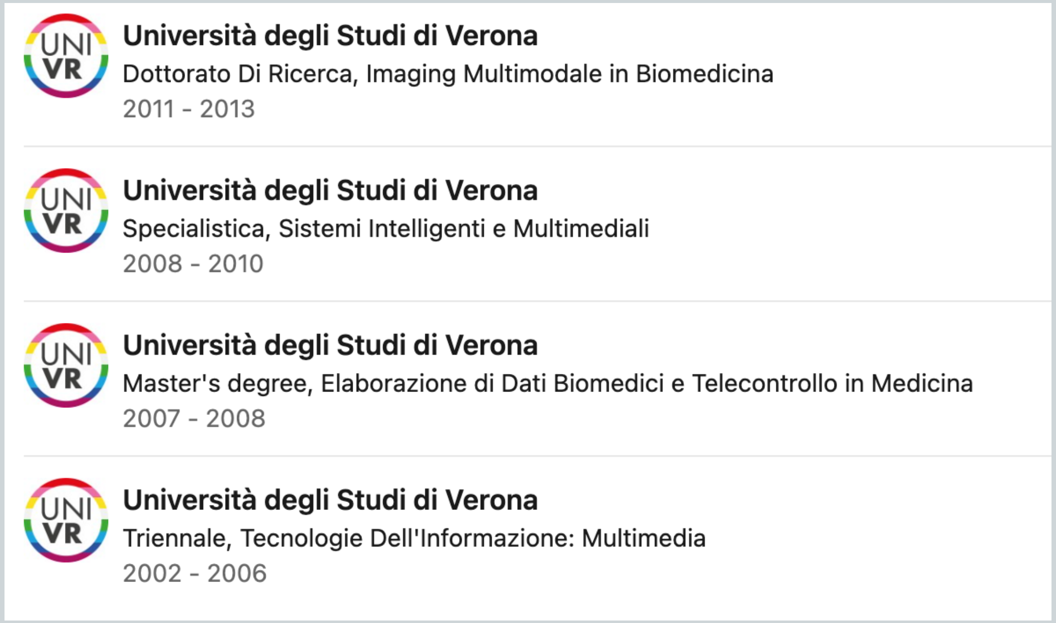
**Research And Development Manager**
AIVIZ Italia
mar 2022 - Presente · 2 anni 1 mese
Verona, Veneto, Italia

**Controllo qualità**

**Computer Vision Engineer**
Brevetti-CEA Spa
mag 2016 - gen 2022 · 5 anni 9 mesi
Vicenza Area, Italy

**Controllo qualità**

**Mobile Software Engineer**
Federico Ferlini Srl
ago 2014 - apr 2016 · 1 anno 9 mesi
Verona Area, Italy



**Università degli Studi di Verona**
Dottorato Di Ricerca, Imaging Multimodale in Biomedicina
2011 - 2013

**Università degli Studi di Verona**
Specialistica, Sistemi Intelligenti e Multimediali
2008 - 2010

**Università degli Studi di Verona**
Master's degree, Elaborazione di Dati Biomedici e Telecontrollo in Medicina
2007 - 2008

**Università degli Studi di Verona**
Triennale, Tecnologie Dell'Informazione: Multimedia
2002 - 2006

## Structure of the course

- Software development skills to create AI applications (L1)
- Python vs C++ (L1)
- Prototyping an AI project in python (frontal lesson)
- Generic and Data Management Packages
- Computer Vision Chapter
- Machine Learning Chapter
- Deep Learning Chapter
- Advanced Vision Applications architectures

**Software development skills to create AI applications**

**Main Phases of Software development**: The development and release of an Artificial Intelligence (AI) application involve several phases, which may vary slightly depending on the context and project specifications. However, the main phases usually include:

- **Requirements Analysis and Problem Definition**:

In this phase, the requirements of the AI application are identified and understood. The problem that the AI will solve is defined, project objectives are determined, and the application domain is established.

- **Data Collection and Preparation**:

Data quality is fundamental to the success of an AI application. In this phase, the necessary data to train the AI model is collected, and data preparation is performed, which may include cleaning, normalization, dimensionality reduction, and anomaly handling.

- **Model Design and Development**:

During this phase, algorithms and machine learning or deep learning models most suitable for solving the identified problem are selected and designed. Models are trained using prepared data, and parameters are optimized to improve performance.

- **Model Evaluation and Validation**:

The trained model is evaluated and validated using separate test data to assess model performance under real conditions. Tests are conducted to measure accuracy, precision, recall, and other relevant metrics.

- **Optimization and Refinement**:

If necessary, the model is optimized and refined to improve performance, reduce training time, optimize resource usage, and address any issues identified during the evaluation phase.

- **Implementation and Integration**:

Once the model is developed and validated, it is integrated into the existing application or a specific AI application is developed. This phase may involve implementing the model in a production environment, integrating with other systems, and configuring the necessary infrastructure for model execution.

- **Monitoring and Maintenance**:

After the application is released, it is important to monitor the model's performance in production and collect feedback from users. Updates, optimizations, and bug fixes can be performed to ensure that the AI application functions efficiently and accurately over time.

- **Release and Distribution**:

Finally, the AI application is released and distributed to end users. This may include releasing on distribution platforms such as app stores, marketplaces, or web platforms, as well as distributing on servers or local devices, depending on the application's needs.

## Real Time vs Non-Real Time

Differences between real-time and non-real-time solutions in the field of Artificial Intelligence (AI) are primarily related to time requirements and application performance. Here are the main differences:

**Non-Real-Time Solutions:**

- Offline Processing: Non-real-time solutions execute data processing in offline or batch mode. This means that data is processed in batches, collected and processed in blocks, without strict time constraints.
- Temporal Flexibility: Non-real-time solutions can operate with longer response times, allowing more time for data processing and AI model training. This enables greater computational complexity and more sophisticated models.
- Job Scheduling: Processing can be scheduled at specific times, such as during the night or during periods of low workload, without the need to adhere to tight time constraints.
- Greater Complexity: Non-real-time solutions can support more complex AI models, trained on a large dataset and using more sophisticated algorithms, as there are no time limitations during processing.

**Real-Time Solutions:**

- **Real-Time Processing**: Real-time solutions must be able to quickly respond to inputs, processing data and generating results within strict time constraints. Responses must be provided in near real-time, within milliseconds or even microseconds.
- **Low Latency**: Real-time solutions require low processing delays to provide timely responses to users or systems that request them. This may require optimization of AI models, complexity reduction, and the use of more efficient processing techniques.
- **Continuous Monitoring**: Real-time solutions often require continuous monitoring of the AI system's performance to ensure it responds promptly and reliably to real-time inputs. Any delays or malfunctions can have critical consequences.
- **Complexity Limitations**: Real-time solutions may be limited by the complexity of AI models and the amount of processed data, as it is necessary to ensure that processing occurs within the specified times.

**Real-Time Solutions:**

- **Real-Time Processing**: Real-time solutions must be able to quickly respond to inputs, processing data and generating results within strict time constraints. Responses must be provided in near real-time, within milliseconds or even microseconds.
- **Low Latency**: Real-time solutions require low processing delays to provide timely responses to users or systems that request them. This may require optimization of AI models, complexity reduction, and the use of more efficient processing techniques.
- **Continuous Monitoring**: Real-time solutions often require continuous monitoring of the AI system's performance to ensure it responds promptly and reliably to real-time inputs. Any delays or malfunctions can have critical consequences.
- **Complexity Limitations**: Real-time solutions may be limited by the complexity of AI models and the amount of processed data, as it is necessary to ensure that processing occurs within the specified times.

## Real-Time Examples (Inference Only)

- **Voice Recognition Systems and Virtual Assistants:** Applications like Apple's Siri, Google Assistant, and Amazon Alexa utilize artificial intelligence for real-time voice recognition. These systems can interpret and respond to users' vocal inputs instantly, providing immediate results such as answering questions, activating smart home devices, or executing voice commands.

- **Real-Time Image and Video Recognition:** Surveillance and security systems can use artificial intelligence for real-time recognition of objects, people, or suspicious behaviors from surveillance videos. These systems can automatically detect and notify emergency situations or unauthorized behaviors.

**Real-Time Examples (Inference Only)**

- **High-Frequency Financial Trading Systems:** In the field of finance, artificial intelligence is used to analyze market data and make real-time trading decisions. High-frequency trading (HFT) systems use AI algorithms to execute financial transactions in milliseconds, exploiting small fluctuations in stock prices.

- **Biometric Signal Recognition:** Biometric authentication systems like facial recognition and fingerprint recognition can use artificial intelligence to quickly identify and verify individuals' identities in real-time. These systems are used for secure access to devices, applications, and services.

- **Industrial Production Monitoring:** In the manufacturing industry, artificial intelligence can be used to monitor and optimize production processes in real-time. AI systems can detect anomalies, predict equipment failures, and optimize workflows to maximize efficiency and product quality.

**Just a kick off session Python VS C++**

Python and C++ are two very different programming languages with different features and purposes. Here are some of the main differences between Python and C++:

- Syntax and Programming Style:
    - Python is known for its clean and readable syntax, which promotes a more concise and intuitive programming style. It is often considered easier to learn and use compared to C++.
    - C++, on the other hand, is a more verbose and complex language compared to Python. It typically requires more attention to detail and a deeper understanding of programming concepts.

- Typing:
    - Python is a dynamically typed language, which means that you do not need to explicitly declare the type of a variable before using it.
    - C++ is a statically typed language, which means that you need to declare the type of each variable before using it, and types are checked at compile time.

**Just a kick off session Python VS C++**

- Memory Management:
  - In Python, memory management is automated through "Garbage Collection," which takes care of freeing memory allocated by objects no longer in use.
  - In C++, you need to manually manage memory allocation and deallocation using pointers and statements like `new` and `delete`.

- Execution Speed:
  - C++ is generally faster in terms of execution compared to Python because the code is compiled directly into machine code rather than being interpreted.
  - Python, being an interpreted language, can be slower in some situations, especially for computationally intensive operations.

**Just a kick off session Python VS C++**

- Usage and Application Domain:
    - Python is often used for rapid application development, prototyping, scripting, web development (using frameworks like Django and Flask), data analysis, machine learning, and many other applications.
    - C++ is often used for applications that require high performance, such as video games, embedded systems, operating systems, low-level applications, and high-level software that require direct control over memory and system resources.

Just a kick off session Python VS C++

- Example C++
- Example Python