**Slide 1 – Title**

Good morning respected faculty and dear classmates.  
My name is Mohammad Alquamah Ansari, and I’m a final-year B.Sc. Artificial Intelligence student at Aditya Degree College, Kakinada.  
Today, I’ll be presenting on how we can scientifically measure software size using COSMIC Full Function Points — or FFP.  
This presentation is based on materials from COSMIC.org, ISO/IEC 19761, and academic case studies.  
Let’s begin by understanding the problem at the heart of software estimation.

**🗣️ Slide 2 – The Core Problem**

Before we can manage a software project, we must first measure its size.  
However, traditional metrics like Lines of Code or Staff-Months do not actually measure functionality — they measure effort or implementation.  
This leads to unscientific and inconsistent estimations.  
The absence of a standard unit for software size is what COSMIC tries to address.

**🗣️ Slide 3 – COSMIC Meta-Model**

The COSMIC Meta-Model gives us a way to measure functional size.  
It’s based on what the software *does* for the user.  
The key components — like the triggering event, functional user, and software boundary — help define the interaction between users and the system.  
This model focuses purely on *functionality*, not how it's implemented.

**🗣️ Slide 4 – Data Movements**

COSMIC uses four types of data movements to measure functionality.  
Entry and Exit represent data moving between the user and system, while Read and Write represent movement between the system and storage.  
Each movement contributes one COSMIC Function Point.  
These form the atomic units of work in the COSMIC method.

**🗣️ Slide 5 – Measurement Process**

COSMIC measurement follows a clear four-step process:  
First, define the scope — what part of the system are we measuring?  
Then, identify all the functional processes, followed by listing each E, X, R, and W data movement.  
Finally, sum them up to get the total number of Function Points.

**🗣️ Slide 6 – Case Study**

Let’s apply COSMIC to a hotel reservation example.  
When a guest books a room, there’s one Entry (data input), one Read (room check), one Write (store reservation), and one Exit (confirmation).  
Each action is one Function Point, giving a total of 4 CFP.  
This shows how COSMIC maps real-world functionality into measurable units.

**🗣️ Slide 7 – Advantages**

COSMIC has several strengths.  
It’s technology-neutral, meaning it doesn’t depend on programming language or platform.  
It’s usable in both business and embedded systems, and it supports benchmarking and outsourcing.  
Because it measures what the system *does*, it offers fair and repeatable comparisons.

**🗣️ Slide 8 – Limitations**

That said, COSMIC isn’t without its limitations.  
It requires detailed specs — vague requirements can’t be measured.  
It also needs expertise to apply correctly and may be time-consuming.  
Compared to traditional function point models, fewer tools support COSMIC — though this is changing.

**🗣️ Slide 9 – From Size to Effort**

COSMIC helps make estimation more scientific by providing a reliable size variable.  
The formula is simple: Effort = Productivity × Size.  
With COSMIC providing the size, project managers can better plan timelines, budgets, and resources.

**🗣️ Slide 10 – Conclusion**

In conclusion, COSMIC Full Function Points give us a formal, objective way to measure software functionality.  
While the method requires training, its benefits — especially for estimation, benchmarking, and contracts — are significant.  
Thank you for your attention. I now welcome any questions you may have.