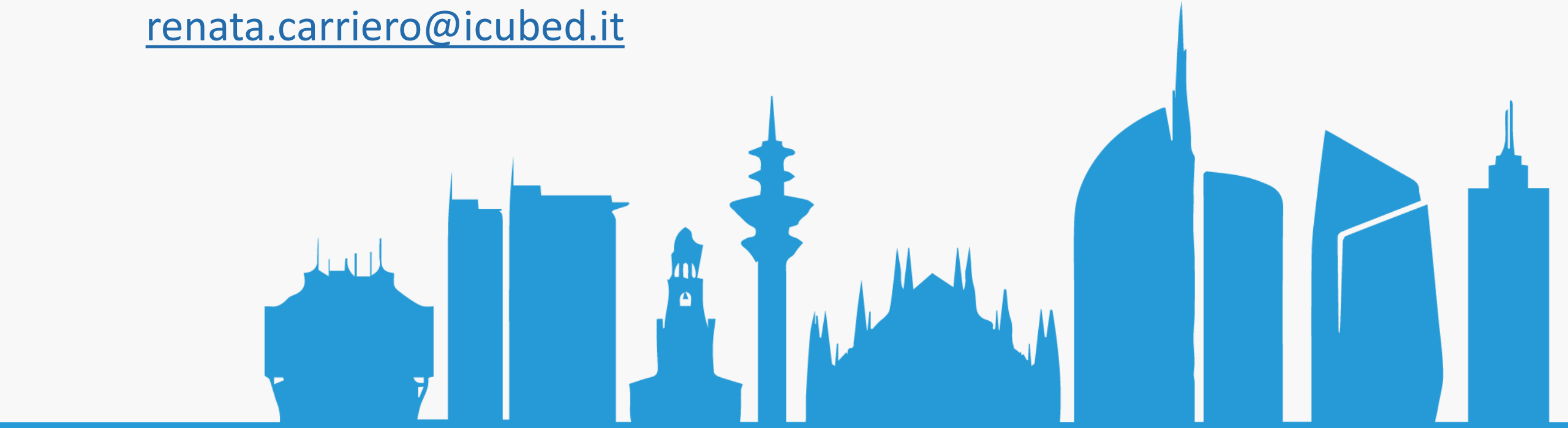
Week 7 – PC Test

Renata Carriero
renata.carriero@icubed.it



Suddivisione del codice

«L'architettura software è l'organizzazione di base di un sistema, espressa dai suoi componenti, dalle relazioni tra di loro e con l'ambiente, e i principi che ne guidano il progetto e l'evoluzione.»

Informalmente, un'architettura software è la **struttura del sistema**, costituita dalle varie **parti** che lo compongono, con le relative **relazioni**.

Suddivisione del codice

L’architettura di un sistema software viene definita nella prima fase di System Design (progettazione architetturale)

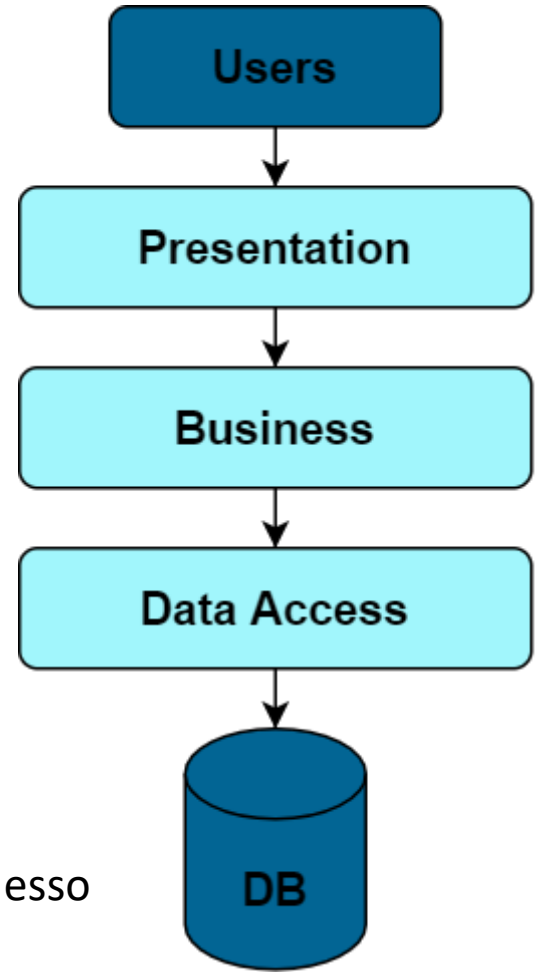
Lo scopo primario è la **scomposizione del sistema in sottosistemi**:

- la realizzazione di più componenti distinti è meno complessa della realizzazione di un sistema come monolito.
- Permette di parallelizzare lo sviluppo
- Favorisce modificabilità, riusabilità, portabilità, etc…

Layer di un Sistema Informativo

A livello concettuale i sistemi informativi
sono progettati in termini di tre diverse
componenti funzionali (livelli)

1. Presentazione (Presentation o User Interface)
2. Logica dell'applicazione (Logica di Business)
3. Gestione delle risorse (Data Access)



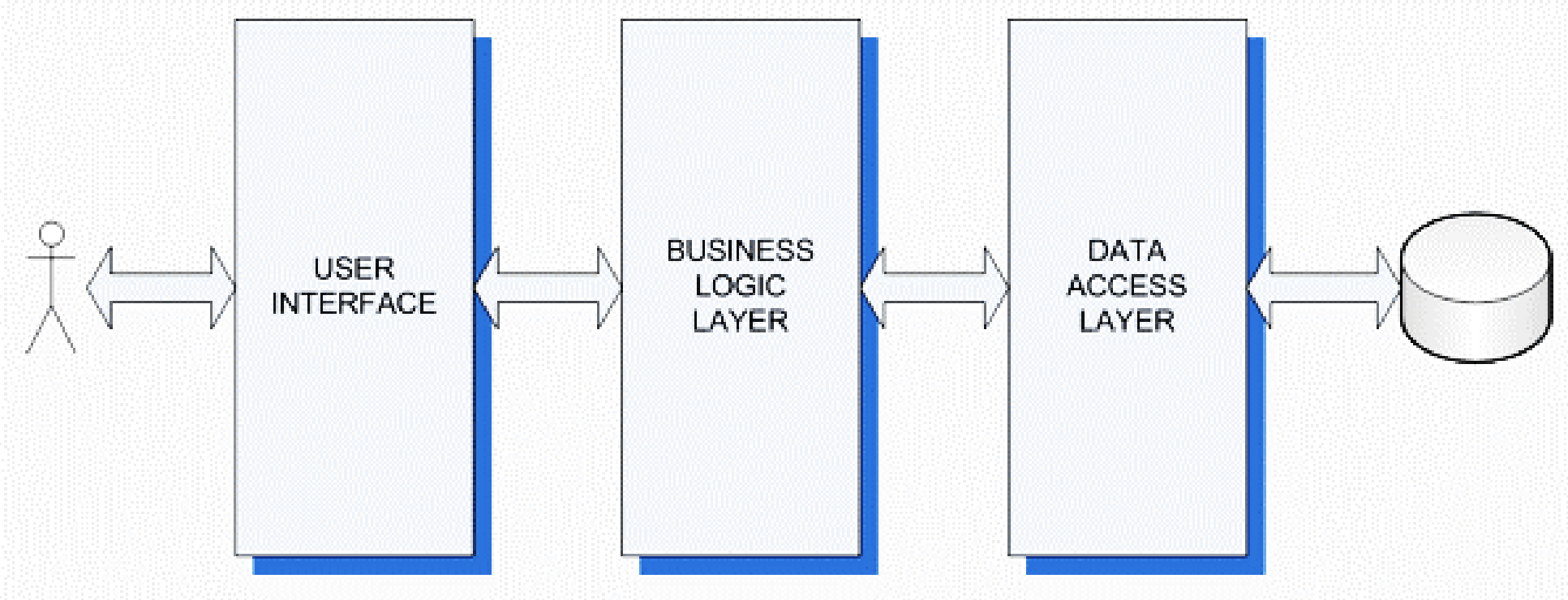
Un layer

- può dipendere solo dai layer di livello più basso
- non ha conoscenza dei layer dei livelli più alti

Architettura chiusa: ogni layer può accedere solo al layer immediatamente sotto di esso

Architettura aperta: un layer può anche accedere ai layer di livello più basso

Layer di un Sistema Informativo



Presentation Layer

- E’ il livello del sistema che gestisce la comunicazione con le entità esterne al sistema stesso (client)

- Comprende le componenti che si occupano di presentare l’informazione verso i client, e che consentono ai client di interagire con il sistema per sottomettere operazioni ed ottenere risultati: **interfaccia utente**.

Application Logic Layer

- E’ il livello del sistema che si occupa del **processamento/elaborazione dei dati** necessario per produrre i risultati da inoltrare al livello di presentazione

  Esempi: Un programma che implementa le operazioni legate ad un prelievo su un conto corrente bancario, o la sequenza di passi da compiere per effettuare un acquisto on-line sono esempi di logica applicativa di un sistema

- Il livello della logica applicativa è anche chiamato **processo di business** o insieme delle regole di business

Data Layer (o Resource Management Layer)

- E’ il livello che **archivia/gestisce i dati** che sono necessari al funzionamento dell’intero sistema

- I dati possono risiedere su una base dati (DB), un file system, o altri contenitori di informazioni

Suddivisione del codice

- Vantaggi:
  - Modo efficiente di condividere grandi moli di dati: «write once for all to read».
  - Un sottosistema non si deve preoccupare di come i dati sono prodotti/usati da ogni altro sottosistema.
  - Gestione centralizzata di backup, security, access control, recovery da errori.
  - Il modello di condivisione dati è pubblicato come repository schema: è molto facile aggiungere nuovi sottosistemi.

- Svantaggi:
  - I sottosistemi devono concordare su un modello dati «di compromesso»: minori performance.
  - «Data evolution». La adozione di un nuovo modello dati è difficile e costosa: deve venir applicato a tutto il repository tutti i sottosistemi devono essere aggiornati.
  - Diversi sottosistemi possono avere diversi requisiti su backup, security e non possono essere supportati con politiche differenti

Suddivisione del codice

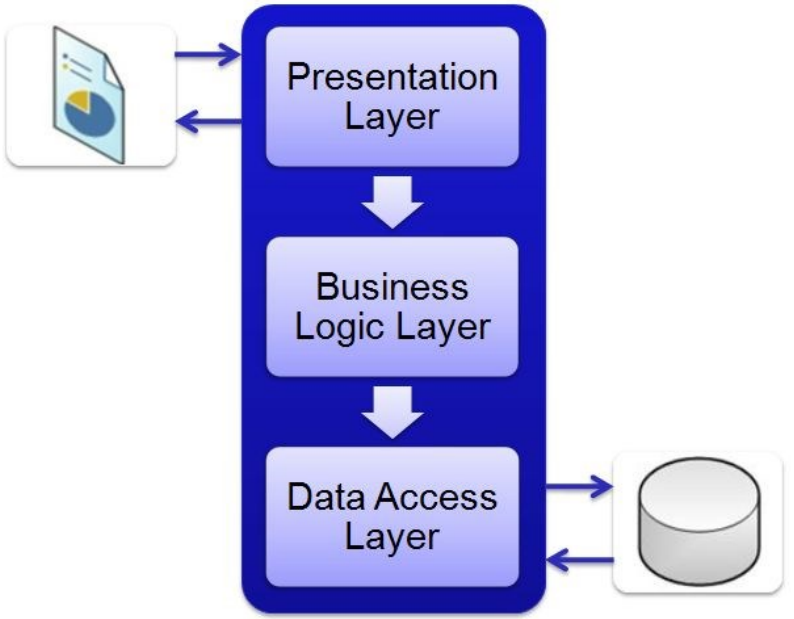
Sono state introdotte all'inizio degli anni '90 per la necessità di definire esplicitamente una business logic che esulasse dalla presentation e dallo storage di un sistema informativo.

Livello 1: gestione dei dati (DBMS, file XML, ecc)

Livello 2: business logic (processamento dati, autenticazione, ecc);

Livello 3: interfaccia utente (presentazione dati, servizi, ecc)

Ogni livello ha obiettivi e vincoli di design propri: nessun livello fa assunzioni sulla struttura o implementazione degli altri ma si limita ad utilizzare le funzioni pubbliche di servizio esposte dagli altri layer per garantire un colloquio a «black box» con essi.



icubed

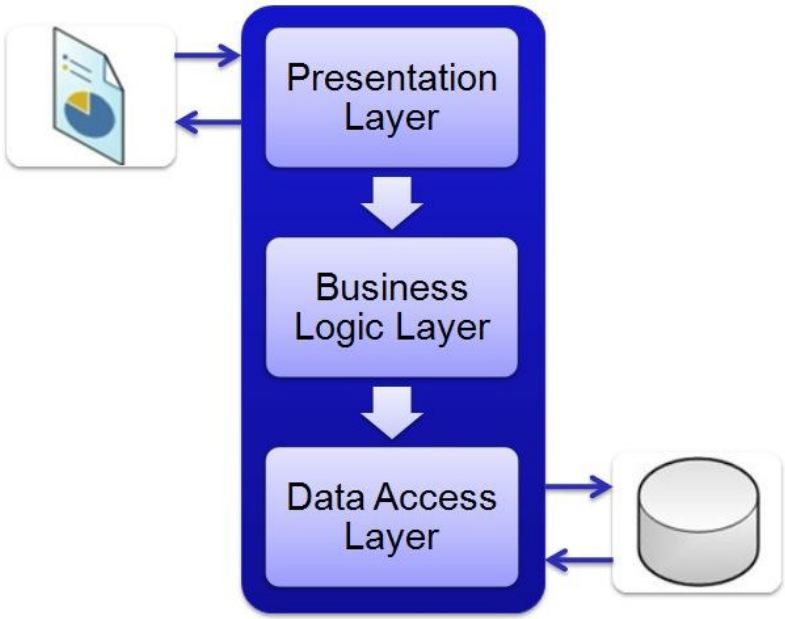
Suddivisione del codice

Non c'è comunicazione diretta tra «livello 1» e «livello 3»:

- L'interfaccia utente non riceve, né inserisce direttamente dati nel livello di data management;
- tutti i passaggi di informazione nei due sensi vengono filtrati dalla business logic

I livelli operano senza assumere di essere parte di una specifica applicazione:

- applicazioni viste come collezioni di componenti cooperanti;
- ogni componente può essere contemporaneamente parte di applicazioni diverse (e.g., database, o componente logica di configurazione di oggetti complessi).



icubed

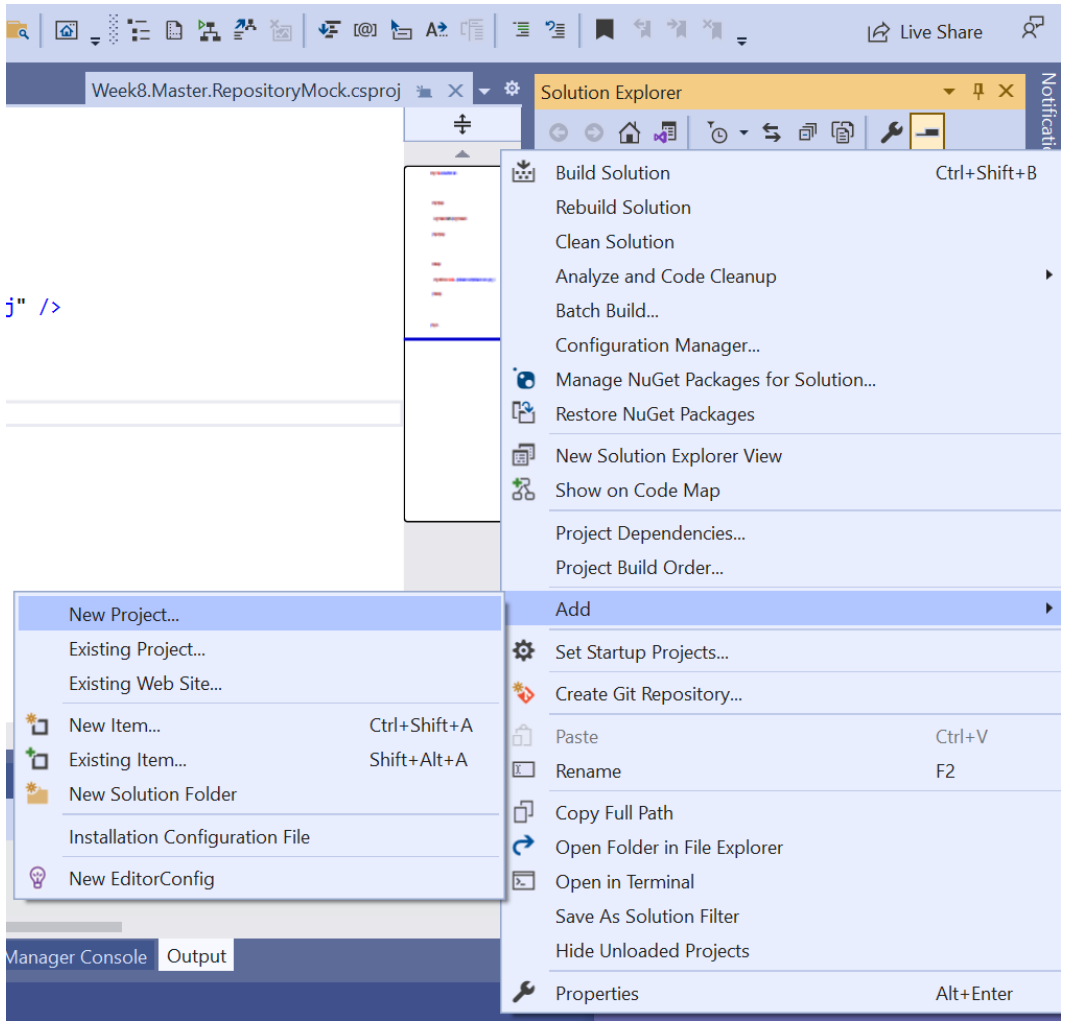
Class Library

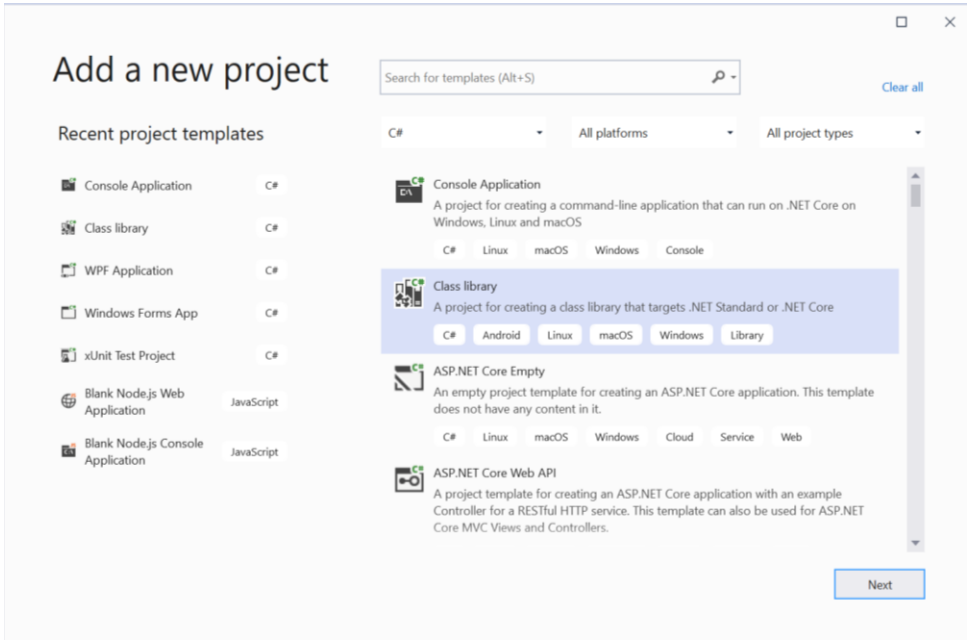
La suddivisione si effettua tramite la creazione di Class Library e la dipendenza tra queste.

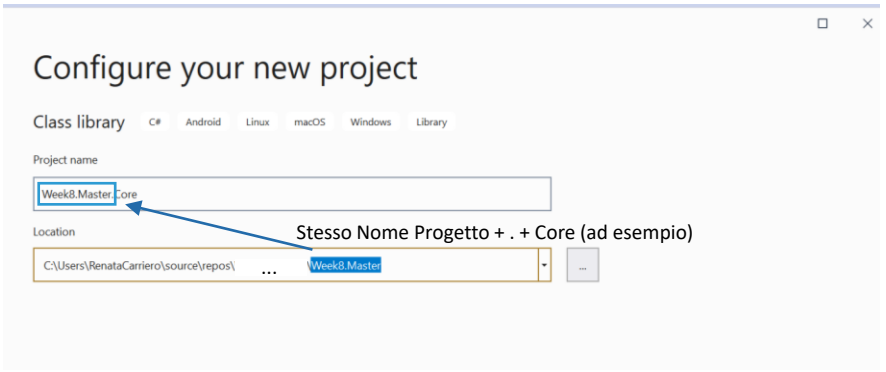
In particolare ci saranno questi 3 livelli:

- **Presentation** (Progetto di tipo App Console oppure WinForm o WPF..)
- **Core** (contiene le Entità, la logica di business e le IRepository)
- **Repository** (Mock o EF ecc.)

Class Library

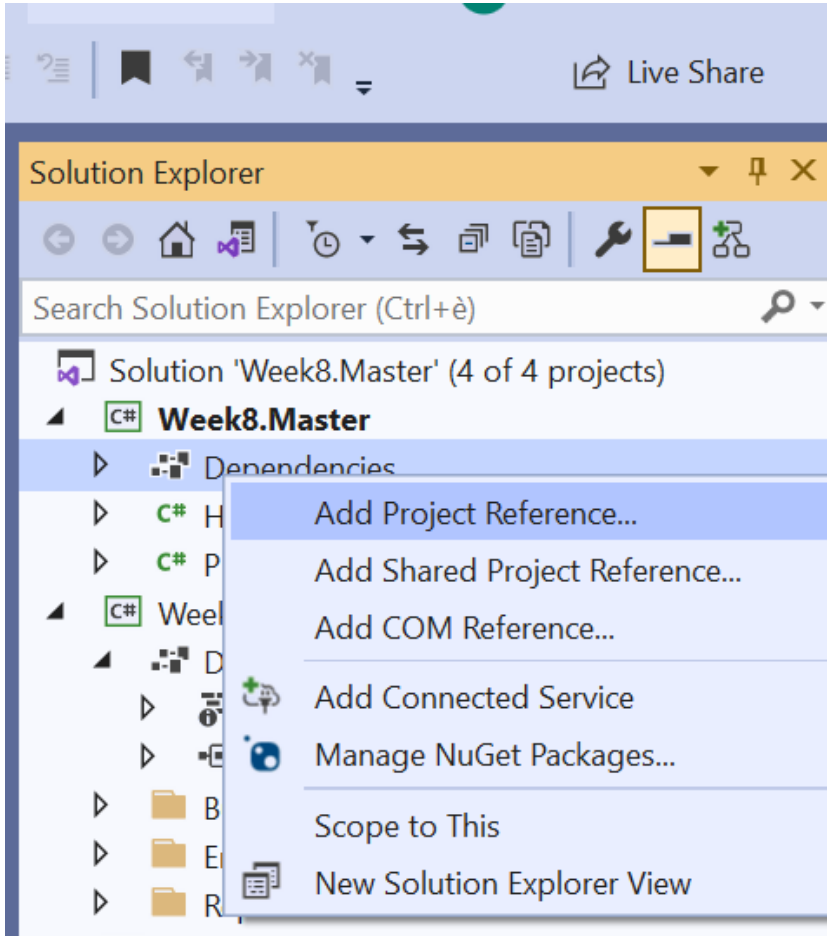


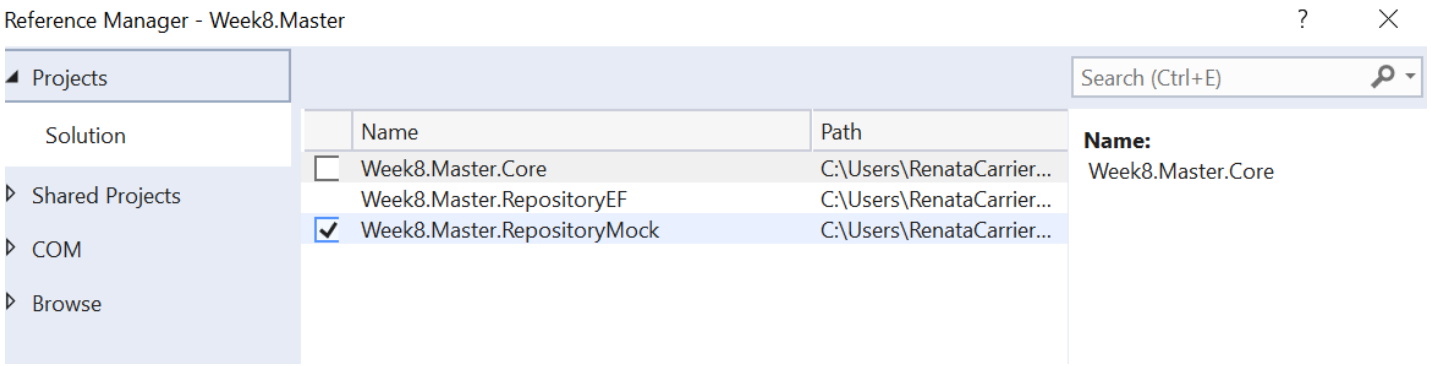


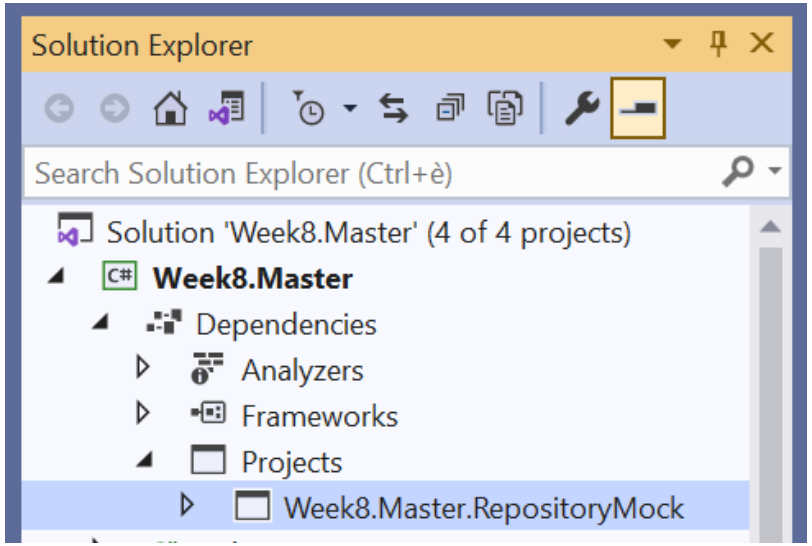


icubed

Dependencies







icubed

Demo

Si vuole realizzare un sistema per la gestione amministrativa dei corsi un Master.

*Un **Corso** è caratterizzato da un codice, un nome e una descrizione.*

*Ciascun corso si articola in un certo numero di Lezioni (almeno una deve avercela)**.***

*I **Docenti** sono caratterizzati da nome, cognome, email e numero di telefono.*
Possono esistere docenti che, per vari motivi, non tengono alcuna lezione.

Ogni Lezione può essere tenuta da un solo docente.
*Ogni **Lezione** ha una data e un orario di inizio, una durata (in termini di giorni) e un'aula assegnata.*

*Ogni corso ha un certo numero di **Studenti** partecipanti per i quali si vuole tenere traccia del nome, cognome, data di nascita, indirizzo email e ultimo titolo di studio.*

Uno studente può partecipare ad un solo corso.

icubed

Demo

Le operazioni da implementare sono:

- *Visualizzazione di tutti i Corsi*
- *Inserisci/Modifica/Elimina Corso*
- *Visualizzazione di tutti i Docenti*
- *Inserisci/Modifica/Elimina Docente (Modifica solo Email e Telefono)*
- *Visualizzazione di tutte le Lezioni*
- *Inserisci/Modifica/Elimina Lezione (Modifica solo Aula)*
- *Visualizza le Lezioni di un Corso ricercando per Codice del Corso*
- *Visualizza le Lezioni di un Corso ricercando per Nome del Corso*
- *Visualizzazione di tutti gli Studenti*
- *Inserisci/Modifica/Elimina Studente (Modifica solo Email)*
- *Visualizzazione Studenti iscritti ad un corso ricercando per Codice del Corso*

NB. Se non specificato, la chiave primaria delle entità è un id intero.

Realizzare il modello E-R della struttura. Utilizzare Entity-Framework per la realizzazione dello strato di persistenza.

icubed