

Architecture of application that uses DMES+API to check records for uniqueness



68 Bridge, St. Suite 304
Suffield, CT 06708



+1 888-779-6578



Sales@DataLadder.com



www.DataLadder.com

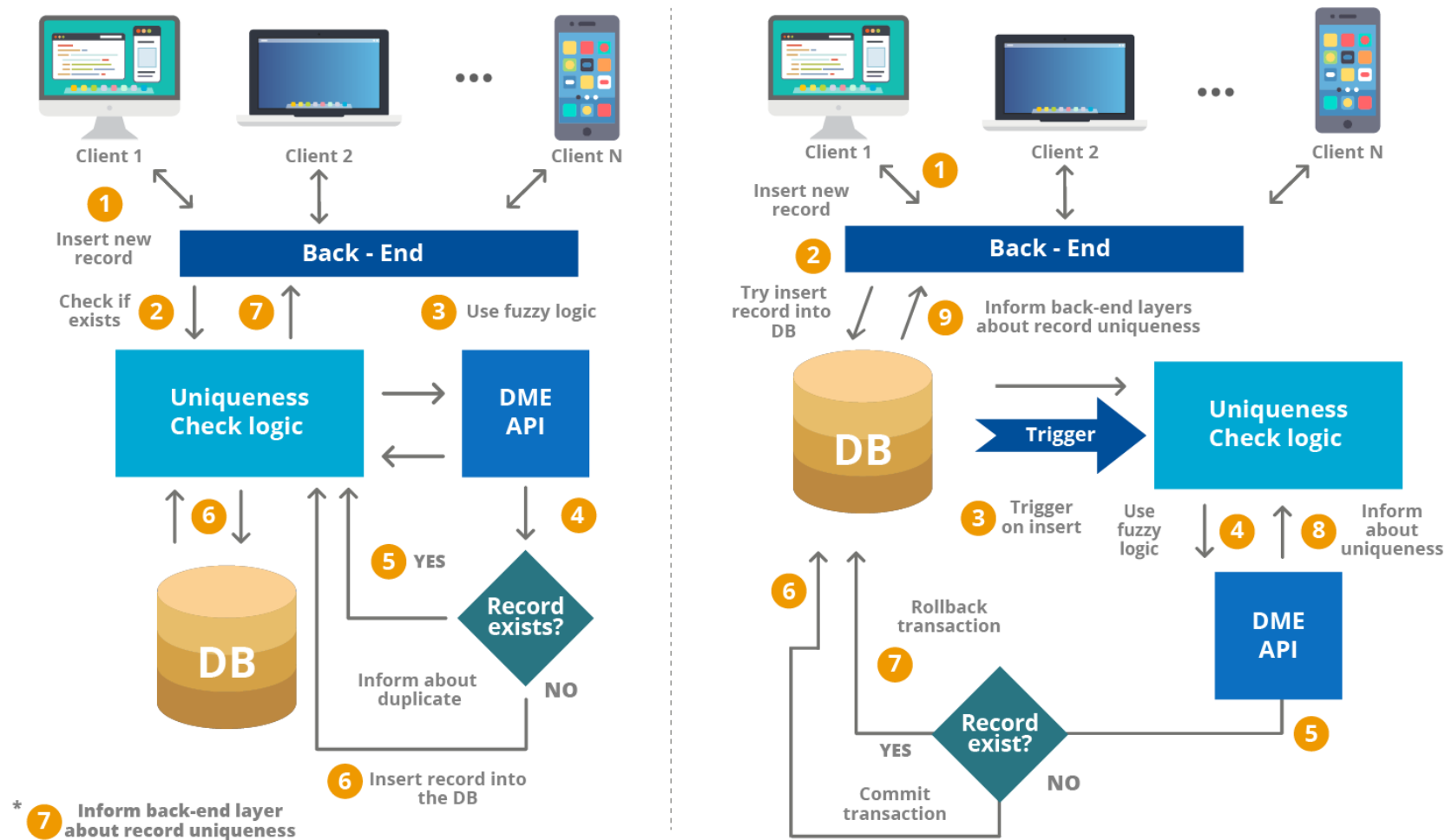


Fig 1. Client server architecture

A) DME API is used as an intermediate layer between DB and a business layer

B) DME API is called from DB triggers

Pros and Cons of different Application Architectures

Approach A:

1. Pros.

- 1.1. Business logic can have a deal with DME API directly. That allow us:
 - 1.1.1. Work with cached data and have higher performance;
 - 1.1.2. Insert only unique records into DB;
 - 1.1.3. Have more flexible control over inserting records.
- 1.2. Less DB load.
- 1.3. Avoiding extra steps of moving data through processing pipeline.

2. Cons.

- 2.1. Uniqueness check logic should be injected into existing communication of Business Layer and DB. It's not always possible due to client's application architecture.
- 2.2. Additional disk space is required for caching data (this disadvantage is related to all DME API usages).

Pros and Cons of different Application Architectures

Approach B:

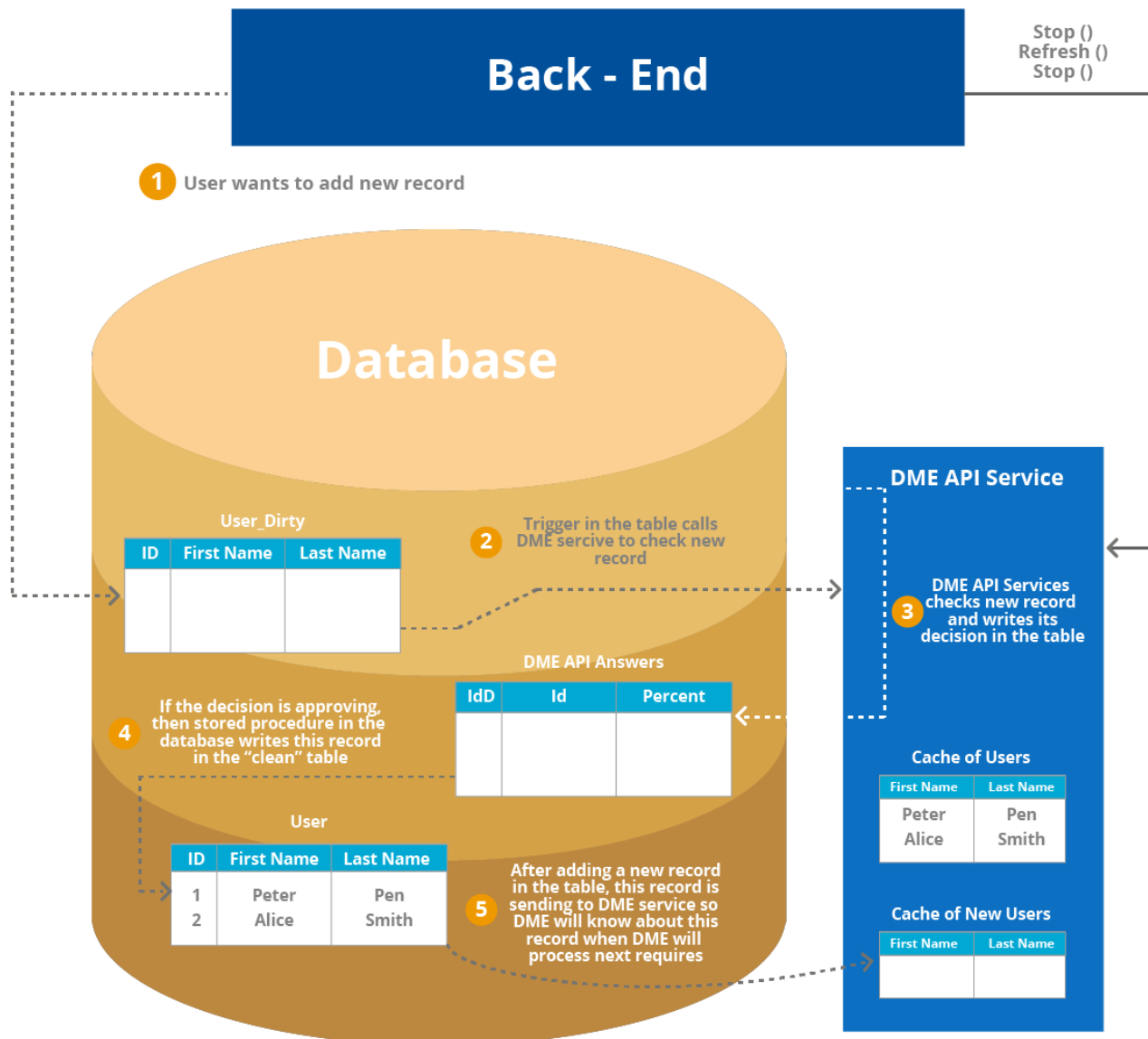
1. Pros.

- 1.1. Adding of uniqueness check module doesn't change existing application infrastructure significantly.
- 1.2. DME API is dealing with cached data, so the speed of record processing should be very high.
- 1.3. If no additional layers could be inserted between Business and Data layers this solution is a single applicable.

2. Cons.

- 2.1. Every time we insert a new record, trigger is called. Also additional operations are needed if the new record is duplicated – rolling back of transactions.
- 2.2. Higher load on DB.
- 2.3. Custom implementation of triggers.
- 2.4. Lower abstraction between Data Layer and Business Layer.

1. State after Start() or Refresh()

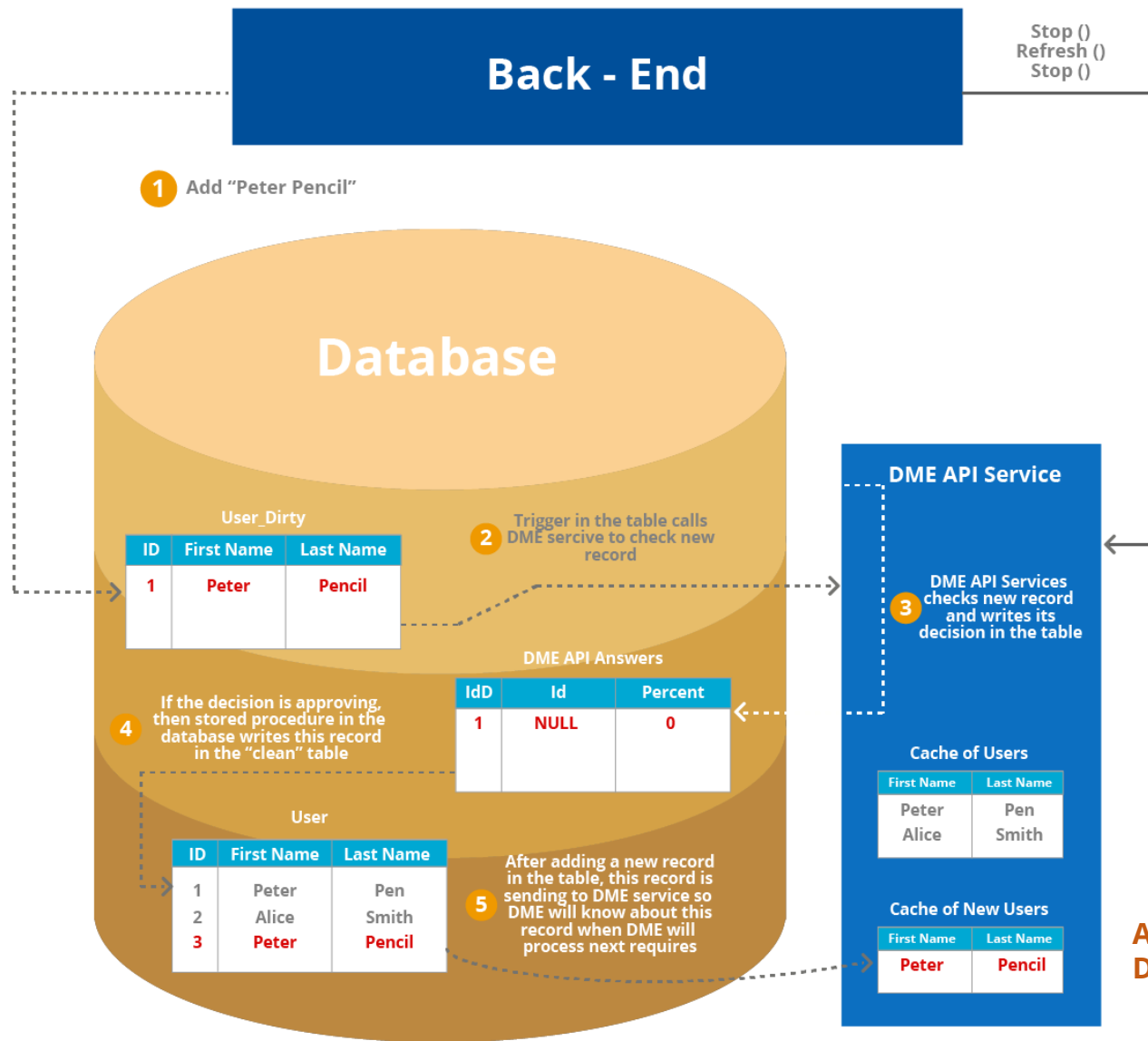


Approach C:

It is similar to approach B – work with DB triggers and stored procedures

After Start() or Refresh() all records are cached by DME API Service in main table

1. State after Start() or Refresh()



Advantages:

- All require from a user are saved in the database. All decisions of DME are saved also.
- This decision decreases a programming in C# from a user and increases programming in SQL (maybe for our users the SQL is more familiar)

Disadvantages:

- Changes in the database
- Increasing a load on database

After Start() or Refresh() all records are cached by DME API Service in main table



Book Demo

Sales@dataladder.com



68 Bridge, St. Suite 304
Suffield, CT 06708



+1 888-779-6578



Sales@DataLadder.com



www.DataLadder.com