This is the group 4 project presentation. Our team chose Multi-Version Concurrency Control Algorithm as our project topic. We are interested in with the large amounts of increasing users, how to efficiently handle the concurrent access to the application. The MVCC algorithm provides an elegant way of solving concurrent Read and Write conflicts without using lock. This is the reason why we chose it as our research topic.

The MVCC algorithm is widely applied on the modern database management system, so our project demo is a functional database system. In the following video, we will briefly introduce the algorithm theory and our database system demo features. (40s)

The concurrently executed transactions would usually have some interleaving read and write operations. This would lead to many unexpected results, such as lost update problems, dirty reads and other inconsistent data retrieval problems. Multi-version concurrency control algorithms solve these problems by managing multiple copies of each data item in the database. When the MVCC database needs to create or update a data item, it produces a newer version of it. Every Read and Write operation would update the timestamp of that item. As a result, each transaction sees a snapshot of the database at a particular instant in time. By intelligently controlling which versions of the data items that different transactions can access, concurrently executed transactions would result in as if they are correctly serialized. Different levels of isolation are therefore achieved. In other words, Read and Write conflicts are solved by version controls instead of read-write locks.

The MVCC algorithm includes two parts, Version Control Algorithm and Concurrency Control Algorithm. The Version Control Algorithm decides when and how to create and update the new version of the data item and how to collect the old version garbage. The Concurrency Control Algorithm gives the standards of which transactions should access which versions of the data item. To see the details of the algorithms, please check the source code and project reports. (60s)

Our project demo is a well-functional database system application. It follows the client and server structure. Database server could take care of multiple client’s concurrent transaction requests. Client users could login and do general database transaction operations. The demo supports Select, Insert, Delete, Update, Commit and Rollback operations. Multi-version concurrency control Algorithm plays a key role at ensuring the correct execution of these transaction operations when multiple clients user concurrently access and update the same data item.

The demo also includes the timeout feature to keep track the validity of a transaction status. We also implement a simple graphical interaction interface for both client-side and server-side to reach a more friendly test and presentation experience. You could try it when running the application. The demo could handle the client process crash in a correct response. To simulate the client crash, we add a clickable crash button on the client interface. The backup module used to keep data storage persistent. To see more details of our project application demo, please check and run the source codes. (60s)

Understanding and implementing the MVCC algorithms is a hard job, developing a fully-functional database system demo is also a big challenge. We are glad that we solve all these challenges and produce a satisfied project outcome at the end.

This is the end of the slide introduction. In the following video, we will show a demonstration of the running MVCC database.  (20s)