

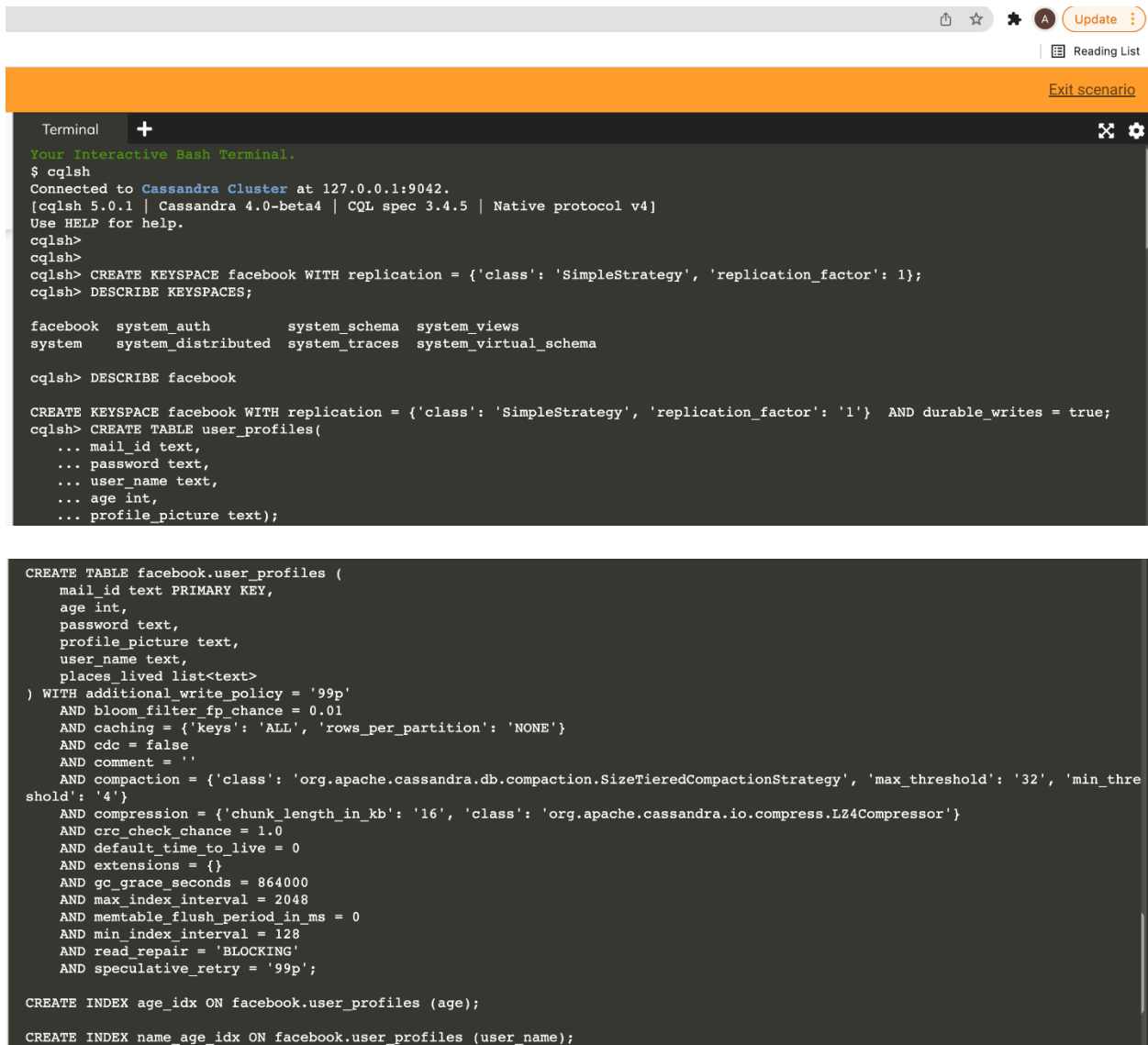
HONORS ASSESSMENT 3

Name - Atharva Paliwal

Roll No. - 40

Sem - 8 Shift-2

Q. Create a facebook like application to store user profile and their social connections, their posts (only text ones for simplicity) and a timeline.



The screenshot shows a web browser window with a terminal interface. The terminal displays the following commands and output:

```
Terminal +
Your Interactive Bash Terminal.
$ cqlsh
Connected to Cassandra Cluster at 127.0.0.1:9042.
[cqlsh 5.0.1 | Cassandra 4.0-beta4 | CQL spec 3.4.5 | Native protocol v4]
Use HELP for help.
cqlsh>
cqlsh>
cqlsh> CREATE KEYSPACE facebook WITH replication = {'class': 'SimpleStrategy', 'replication_factor': 1};
cqlsh> DESCRIBE KEYSPACES;

facebook  system_auth          system_schema  system_views
system    system_distributed  system_traces  system_virtual_schema

cqlsh> DESCRIBE facebook

CREATE KEYSPACE facebook WITH replication = {'class': 'SimpleStrategy', 'replication_factor': 1} AND durable_writes = true;
cqlsh> CREATE TABLE user_profiles(
... mail_id text,
... password text,
... user_name text,
... age int,
... profile_picture text);

CREATE TABLE facebook.user_profiles (
  mail_id text PRIMARY KEY,
  age int,
  password text,
  profile_picture text,
  user_name text,
  places_lived list<text>
) WITH additional_write_policy = '99p'
AND bloom_filter_fp_chance = 0.01
AND caching = {'keys': 'ALL', 'rows_per_partition': 'NONE'}
AND cdc = false
AND comment = ''
AND compaction = {'class': 'org.apache.cassandra.db.compaction.SizeTieredCompactionStrategy', 'max_threshold': '32', 'min_threshold': '4'}
AND compression = {'chunk_length_in_kb': '16', 'class': 'org.apache.cassandra.io.compress.LZ4Compressor'}
AND crc_check_chance = 1.0
AND default_time_to_live = 0
AND extensions = {}
AND gc_grace_seconds = 864000
AND max_index_interval = 2048
AND memtable_flush_period_in_ms = 0
AND min_index_interval = 128
AND read_repair = 'BLOCKING'
AND speculative_retry = '99p';

CREATE INDEX age_idx ON facebook.user_profiles (age);

CREATE INDEX name_age_idx ON facebook.user_profiles (user_name);
```

```
cqlsh:facebook> CREATE TABLE user_profiles( mail_id text, password text, user_name text, age int, profile_picture text, PRIMARY KEY(mail_id));
cqlsh:facebook> INSERT INTO user_profiles(mail_id, password, user_name, age, profile_picture)
... VALUES('monica@gmail.com', 'password', 'monica', 21, 'monica.jpg');
cqlsh:facebook> INSERT INTO user_profiles(mail_id, password, user_name, age, profile_picture)
... VALUES('abc@gmail.com', 'password', 'abc', 21, 'abc.jpg');
cqlsh:facebook> INSERT INTO user_profiles(mail_id, password, user_name, age, profile_picture)
... VALUES('user1@gmail.com', 'password', 'user1', 21, 'user1.jpg');
cqlsh:facebook> INSERT INTO user_profiles(mail_id, password, user_name, age, profile_picture)
... VALUES('user2@gmail.com', 'password', 'user2', 21, 'user2.jpg');
cqlsh:facebook> SELECT * from user_profiles;
```

mail_id	age	password	profile_picture	user_name
abc@gmail.com	21	password	abc.jpg	abc
user2@gmail.com	21	password	user2.jpg	user2
user1@gmail.com	21	password	user1.jpg	user1

```
cqlsh:facebook> CREATE INDEX IF NOT EXISTS name_idx on facebook.user_profiles(user_name);
cqlsh:facebook> CREATE INDEX IF NOT EXISTS age_idx on facebook.user_profiles (age);
```

```
(1 rows)
cqlsh:facebook> select * from user_profiles where user_name = 'user1' and age>20 ALLOW FILTERING;
```

mail_id	age	password	profile_picture	user_name
user1@gmail.com	21	password	user1.jpg	user1

```
(1 rows)
cqlsh:facebook> ALTER TABLE user_profiles
... ADD places_lived list<text>;
```

```
cqlsh:facebook> CREATE TABLE posts(
... id int PRIMARY KEY,
... actual_post text,
... liked_by set<text>,
... comments map<text,text>);
cqlsh:facebook> DESCRIBE posts;
```

```
CREATE TABLE facebook.posts (
  id int PRIMARY KEY,
  actual_post text,
  comments map<text, text>,
  liked_by set<text>
) WITH additional_write_policy = '99p'
AND bloom_filter_fp_chance = 0.01
AND caching = {'keys': 'ALL', 'rows_per_partition': 'NONE'}
AND cdc = false
AND comment = ''
AND compaction = {'class': 'org.apache.cassandra.db.compaction.SizeTieredCompactionStrategy', 'max_threshold': '32', 'min_threshold': '4'}
AND compression = {'chunk_length_in_kb': '16', 'class': 'org.apache.cassandra.io.compress.LZ4Compressor'}
AND crc_check_chance = 1.0
AND default_time_to_live = 0
AND extensions = {}
AND gc_grace_seconds = 864000
AND max_index_interval = 2048
AND memtable_flush_period_in_ms = 0
AND min_index_interval = 128
AND read_repair = 'BLOCKING'
AND speculative_retry = '99p';
```

```
cqlsh:facebook> INSERT INTO posts(id, actual_post, liked_by, comments) values(1, 'Hey, I am on Facebook now!', {'monica','abc','user1'}, {'abc':'Welcome to the facebook family', 'user1': 'Welcome!'});
cqlsh:facebook> SELECT * from posts;
```

id	actual_post	comments	liked_by
1	Hey, I am on Facebook now!	{'abc': 'Welcome to the facebook family', 'user1': 'Welcome!'}	{'abc', 'monica', 'user1'}

```
(1 rows)
cqlsh:facebook> UPDATE posts
... SET liked_by=liked_by+{'user2'}
... WHERE id=1;
cqlsh:facebook> SELECT * from posts;
```

id	actual_post	comments	liked_by
1	Hey, I am on Facebook now!	{'abc': 'Welcome to the facebook family', 'user1': 'Welcome!'}	{'abc', 'monica', 'user1', 'user2'}

```
cqlsh:facebook> CREATE TABLE timeline(
... mail_id text,
... post_id int,
... body text,
... user_name text,
... device text,
... posted timestamp,
... liked_by set<text>,
... comments map<text, text>,
... PRIMARY KEY(mail_id, post_id));
```

```
cqlsh:facebook> INSERT INTO timeline (mail_id, post_id, body, user_name, device, posted, liked_by, comments) VALUES('user1@gmail.com', 1, 'Hey, there!', 'user1', 'mobile', '2022-02-18', {'abc', 'user1'}, {'abc': 'Hello, user1'});
cqlsh:facebook> SELECT * from timeline;
```

mail_id user_name	post_id	body	comments	device	liked_by	posted
user1@gmail.com user1	1	Hey, there!	{'abc': 'Hello, user1'}	mobile	{'abc', 'user1'}	2022-02-18 00:00:00.000000+0000

```
(1 rows)
cqlsh:facebook> INSERT INTO timeline (mail_id, post_id, body, user_name, device, posted, liked_by, comments) VALUES('user2@gmail.com', 1, 'Hi, this is user2!', 'user2', 'mobile', '2022-02-19', {'abc', 'user1', 'user2'}, {'abc': 'Hello, user2'});
cqlsh:facebook> SELECT * FROM timeline;
```

mail_id user_name	post_id	body	comments	device	liked_by	posted
user2@gmail.com 00.000000+0000 user1@gmail.com 00.000000+0000	1 user2 1 user1	Hi, this is user2! Hey, there!	{'abc': 'Hello, user2'} {'abc': 'Hello, user1'}	mobile	{'abc', 'user1', 'user2'} {'abc', 'user1'}	2022-02-19 00:00:00.000000+0000 2022-02-18 00:00:00.000000+0000

```
(2 rows)
cqlsh:facebook> SELECT * FROM timeline
... WHERE mail_id='user1@gmail.com' and post_id=1;
```

```
cqlsh:facebook> SELECT * FROM timeline WHERE mail_id='user1@gmail.com' and post_id=1;
```

mail_id user_name	post_id	body	comments	device	liked_by	posted
user1@gmail.com user1	1	Hey, there!	{'abc': 'Hello, user1'}	mobile	{'abc', 'user1'}	2022-02-18 00:00:00.000000+0000

```
(1 rows)
```

```
cqlsh:facebook> CREATE TABLE friends(
    ... mail_id text,
    ... friend_mail_id text,
    ... friends_since timestamp,
    ... subscribe_to_post int,
    ... PRIMARY KEY(mail_id, friend_mail_id));
cqlsh:facebook> DESCRIBE friends;

CREATE TABLE facebook.friends (
    mail_id text,
    friend_mail_id text,
    friends_since timestamp,
    subscribe_to_post int,
    PRIMARY KEY (mail_id, friend_mail_id)
) WITH CLUSTERING ORDER BY (friend_mail_id ASC)
    AND additional_write_policy = '99p'
    AND bloom_filter_fp_chance = 0.01
    AND caching = {'keys': 'ALL', 'rows_per_partition': 'NONE'}
    AND cdc = false
    AND comment = ''
    AND compaction = {'class': 'org.apache.cassandra.db.compaction.SizeTieredCompactionStrategy', 'max_threshold': '32', 'min_thre
shold': '4'}
    AND compression = {'chunk_length_in_kb': '16', 'class': 'org.apache.cassandra.io.compress.LZ4Compressor'}
    AND crc_check_chance = 1.0
    AND default_time_to_live = 0
    AND extensions = {}
    AND gc_grace_seconds = 864000
    AND max_index_interval = 2048
    AND memtable_flush_period_in_ms = 0
    AND min_index_interval = 128
    AND read_repair = 'BLOCKING'
    AND speculative_retry = '99p';
```

```
cqlsh:facebook> INSERT INTO friends(mail_id, friend_mail_id, friends_since, subscribe_to_post)
    ... VALUES('user1@gmail.com', 'user2@gmail.com', '2022-02-18', 1);
cqlsh:facebook> INSERT INTO friends(mail_id, friend_mail_id, friends_since, subscribe_to_post)
    ... VALUES('user2@gmail.com', 'user1@gmail.com', '2022-02-18', 1);
cqlsh:facebook> SELECT * FROM friends;
```

mail_id	friend_mail_id	friends_since	subscribe_to_post
user2@gmail.com	user1@gmail.com	2022-02-18 00:00:00.000000+0000	1
user1@gmail.com	user2@gmail.com	2022-02-18 00:00:00.000000+0000	1

(2 rows)

```
cqlsh:facebook> CREATE TABLE counter(
    ... mail_id text,
    ... user_name text,
    ... friends_count counter,
    ... posts_cout counter,
    ... PRIMARY KEY(mail_id, user_name));
cqlsh:facebook> DESCRIBE COUNTER;

CREATE TABLE facebook.counter (
    mail_id text,
    user_name text,
    friends_count counter,
    posts_cout counter,
    PRIMARY KEY (mail_id, user_name)
) WITH CLUSTERING ORDER BY (user_name ASC)
    AND additional_write_policy = '99p'
    AND bloom_filter_fp_chance = 0.01
    AND caching = {'keys': 'ALL', 'rows_per_partition': 'NONE'}
    AND cdc = false
    AND comment = ''
    AND compaction = {'class': 'org.apache.cassandra.db.compaction.SizeTieredCompactionStrategy', 'max_threshold': '32', 'min_thre
shold': '4'}
    AND compression = {'chunk_length_in_kb': '16', 'class': 'org.apache.cassandra.io.compress.LZ4Compressor'}
    AND crc_check_chance = 1.0
    AND default_time_to_live = 0
    AND extensions = {}
    AND gc_grace_seconds = 864000
    AND max_index_interval = 2048
    AND memtable_flush_period_in_ms = 0
    AND min_index_interval = 128
    AND read_repair = 'BLOCKING'
    AND speculative_retry = '99p';
```

```
cqlsh:facebook> UPDATE counter
... SET friends_count=friends_count+1
... WHERE mail_id='user1@gmail.com' and user_name='user1';
cqlsh:facebook> SELECT * FROM counter;
```

mail_id	user_name	friends_count	posts_cout
user1@gmail.com	user1	1	null

(1 rows)

```
cqlsh:facebook> UPDATE counter
... SET friends_count=friends_count+1
... WHERE mail_id='user1@gmail.com' and user_name='user1';
cqlsh:facebook> UPDATE counter
... SET posts_count=posts_count+1
... WHERE mail_id='user1@gmail.com' and user_name='user1';
```

```
cqlsh:facebook> UPDATE counter
... SET posts_cout=posts_cout+1
... WHERE mail_id='user1@gmail.com' and user_name='user1';
cqlsh:facebook> SELECT * FROM counter;
```

mail_id	user_name	friends_count	posts_cout
user1@gmail.com	user1	2	1

(1 rows)

```
cqlsh:facebook> █
```