System Requirement Specification (SRS)

All the system requirements have been installed in the node_module folder. Thus, the project can be directly run using `node app.js`.

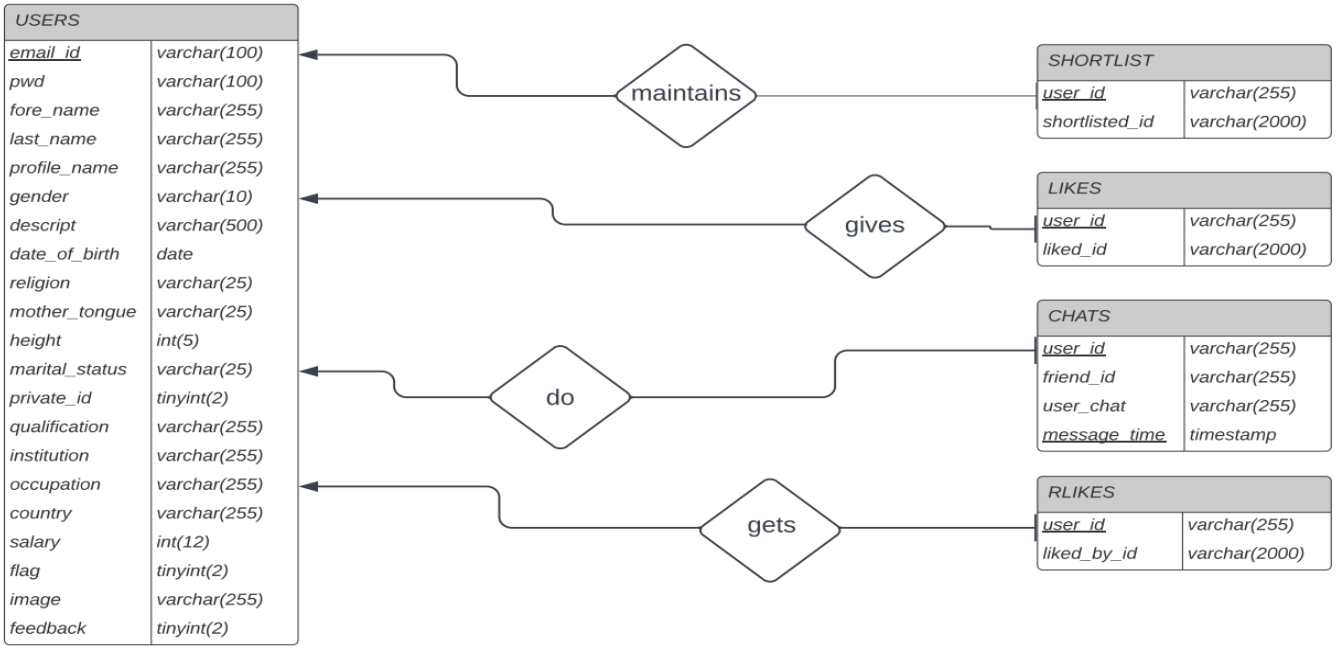
System Requirements:
express
path
fs
express-session
moment
body-parser
split-string
querystring
mysql2
ejs
multer
express-fileupload
dialog
underscore

Other System Requirements:
Mysql
Any IDE(intellij idea)
Web browser(prefer chrome)

Steps to be followed:
In the config.js add your mysql password. Next run the .sql file to create the database. After downloading all the requirements run the code.

System Modelling

❖ Entity-relationship (ER) diagram



❖ Schema design

users(email_id, pwd, fore_name, last_name, profile_name, gender, descript, date_of_birth, religion, mother_tongue, height, marital_status, private_id, qualification, institution, occupation, country, salary, flag, image, feedback)

shortlist(user_id, shortlisted_id)

likes(user_id, liked_id)

rlikes(user_id, liked_by_id)

chats(user_id, friend_id, user)chat, message_time)

maintains(email_id, user_id)

gives(email_id, user_id)

do(email_id, user_id, message_time)

gets(email_id, user_id)

❖ Data normalization

The data we used is present in the normalized form (BCNF). In the users, shortlist, likes, rlikes tables only single attribute can act as the candidate key. So, we have only one candidate key which is selected as the primary key. Coming to the chats table, it has tuple of user_id and message_time as the primary key. So, we need to check whether it is making the whole database in 3NF or not.
If primary key is a single attribute, then it is definitely a candidate key (Ck). For all functional dependencies, the determinant is a Ck, thus the data is in BCNF.
As no proper subset of the primary key don't imply to any other non-prime attribute, thus there is no partial dependencies present. So, it is in 2NF. Also, a non-prime attribute doesn't imply to another non-prime attribute, thus there is no transitive dependencies present. So, it is in 3NF also. This is all obvious from the fact that the data is in BCNF, and thus in 3NF, 2NF, 1NF inherently.

Hence the data is normalized and is present in BCNF.

❖ List of tables required

- users
- shortlist
- likes
- rlikes
- chats

❖ Additional components

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`dashboard`(IN email VARCHAR(255))
READS SQL DATA
DETERMINISTIC
SQL SECURITY INVOKER
COMMENT 'CHECK EXISTENCE OF LOGIN CREDENTIALS'
BEGIN
select *
from users
where not email_id = email;
END$$
DELIMITER ;

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`dashboard_preference_1`(IN email VARCHAR(255))
READS SQL DATA
DETERMINISTIC
SQL SECURITY INVOKER
COMMENT 'GIVE ALL SHORTLISTED EMAIL_ID OF GIVEN
EMAIL_ID'
BEGIN
select shortlisted_id
from shortlist
where user_id=email;
END$$
DELIMITER ;
```

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`dashboard_preference_2`(IN email VARCHAR(255))
READS SQL DATA
DETERMINISTIC
SQL SECURITY INVOKER
COMMENT 'GIVE ALL EMAIL_ID LIKED BY GIVEN
EMAIL_ID'
BEGIN
select liked_id
from likes
where user_id=email;
END$$
DELIMITER ;

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`dashboard_preference_3`(IN nGender
VARCHAR(10),nStartAge INT,nEndAge INT,nReligion
VARCHAR(25),nMotherTongue VARCHAR(25))
READS SQL DATA
DETERMINISTIC
SQL SECURITY INVOKER
COMMENT 'GIVE ALL USERS MATCHING THE INPUTS'
BEGIN
select *
from users
where gender=nGender and year(curdate())-
year(date_of_birth) >= nStartAge
and year(curdate())-year(date_of_birth) <= nEndAge and
religion=nReligion and mother_tongue=nMotherTongue;
END$$
DELIMITER ;
```

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`dashboard_preference_4`(IN nGender
VARCHAR(10),nStartAge INT,nEndAge INT,nReligion
VARCHAR(25),nMotherTongue VARCHAR(25), emailList
VARCHAR(2000))
READS SQL DATA
DETERMINISTIC
SQL SECURITY INVOKER
COMMENT 'GIVE ALL USERS MATCHING THE INPUTS BUT
NOT MATCHING THE EMAILLIST'
BEGIN
select *
from users
where gender=nGender and year(curdate())-
year(date_of_birth) >= nStartAge
and year(curdate())-year(date_of_birth) <= nEndAge and
religion=nReligion and mother_tongue=nMotherTongue
and not FIND_IN_SET(email_id, emailList);
END$$
DELIMITER ;

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`dashboard_userProfile_shortList_1`(IN email
VARCHAR(255))
READS SQL DATA
DETERMINISTIC
SQL SECURITY INVOKER
COMMENT 'GIVE ALL SHORTLISTED EMAIL_ID OF GIVEN
EMAIL_ID'
BEGIN
select shortlisted_id
```

```
from shortlist
where user_id=email;
END$$
DELIMITER ;

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`dashboard_userProfile_shortList_2`(IN email
VARCHAR(255))
READS SQL DATA
DETERMINISTIC
SQL SECURITY INVOKER
COMMENT 'GIVE ALL EMAIL_ID LIKED BY GIVEN
EMAIL_ID'
BEGIN
select liked_id
from likes
where user_id=email;
END$$
DELIMITER ;

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`dashboard_userProfile_shortList_3`(IN  shortEmailList
VARCHAR(2000), likeEmailList VARCHAR(2000))
READS SQL DATA
DETERMINISTIC
SQL SECURITY INVOKER
COMMENT 'GIVE ALL USERS PRESENT IN SHORTEMAILLIST
BUT NOT PRESENT IN LIKEEMAILLIST'
BEGIN
select *
from users
```

```
where FIND_IN_SET(email_id, shortEmailList) AND not
FIND_IN_SET(email_id, likeEmailList);
END$$
DELIMITER ;

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`dashboard_userProfile_yourLikes_1`(IN email
VARCHAR(255))
READS SQL DATA
DETERMINISTIC
SQL SECURITY INVOKER
COMMENT 'GIVE ALL EMAIL_ID LIKED BY GIVEN
EMAIL_ID'
BEGIN
select liked_id
from likes
where user_id=email;
END$$
DELIMITER ;

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`dashboard_userProfile_yourLikes_2`(IN  likeEmailList
VARCHAR(2000))
READS SQL DATA
DETERMINISTIC
SQL SECURITY INVOKER
COMMENT 'GIVE ALL USERS PRESENT IN LIKEEMAILLIST'
BEGIN
select *
from users
where FIND_IN_SET(email_id, likeEmailList);
```

```
END$$
DELIMITER ;

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`dashboard_userProfile_likedBy_1`(IN email
VARCHAR(255))
READS SQL DATA
DETERMINISTIC
SQL SECURITY INVOKER
COMMENT 'GIVE ALL EMAIL_ID THAT LIKED GIVEN
EMAIL_ID'
BEGIN
select liked_by_id
from rlikes
where user_id=email;
END$$
DELIMITER ;

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`dashboard_userProfile_likedBy_2`(IN  likeByEmailList
VARCHAR(2000))
READS SQL DATA
DETERMINISTIC
SQL SECURITY INVOKER
COMMENT 'GIVE ALL USERS PRESENT IN
LIKEBYEMAILLIST'
BEGIN
select *
from users
where FIND_IN_SET(email_id, likeByEmailList);
END$$
```

```
DELIMITER ;

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`shortList_like_1`(IN email VARCHAR(255))
READS SQL DATA
DETERMINISTIC
SQL SECURITY INVOKER
COMMENT 'GIVE INFO OF ALL EMAIL_ID LIKED BY GIVEN
EMAIL_ID'
BEGIN
select *
from likes
where user_id=email;
END$$
DELIMITER ;

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`shortList_like_2`(IN likedEmail VARCHAR(2000),email
VARCHAR(255))
READS SQL DATA
DETERMINISTIC
SQL SECURITY INVOKER
COMMENT 'UPDATE LIKED_ID AT EMAIL WITH
LIKEDEMAIL'
BEGIN
UPDATE likes SET liked_id = likedEmail where
user_id=email;
END$$
DELIMITER ;
```

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`shortList_like_3`(IN likedEmail VARCHAR(2000),email
VARCHAR(255))
READS SQL DATA
DETERMINISTIC
SQL SECURITY INVOKER
COMMENT 'INSERT NEW EMAIL ADN ITS LIKEDEMAIL'
BEGIN
INSERT INTO likes values(email,likedEmail);
END$$
DELIMITER ;

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`shortList_delete`(IN email VARCHAR(255))
READS SQL DATA
DETERMINISTIC
SQL SECURITY INVOKER
COMMENT 'DELETE DATA FROM SHORTLIST'
BEGIN
DELETE FROM shortlist where user_id = email;
END$$
DELIMITER ;

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`dashboard_userProfile_delete`(IN email VARCHAR(255))
READS SQL DATA
DETERMINISTIC
SQL SECURITY INVOKER
COMMENT 'DELETE DATA FROM USER'
BEGIN
```

```
DELETE FROM users where email_id = email;
END$$
DELIMITER ;

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`love_1`(IN email VARCHAR(255))
READS SQL DATA
DETERMINISTIC
SQL SECURITY INVOKER
COMMENT 'GIVE INFO OF ALL EMAIL_ID THAT IS
SHORTLISTED BY GIVEN EMAIL_ID'
BEGIN
select *
from shortlist
where user_id=email;
END$$
DELIMITER ;

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`love_2`(IN shortlistedEmail VARCHAR(2000),email
VARCHAR(255))
READS SQL DATA
DETERMINISTIC
SQL SECURITY INVOKER
COMMENT 'UPDATE SHORTLISTED_ID AT EMAIL WITH
SHORTLISTEDEMAIL'
BEGIN
UPDATE shortlist SET shortlisted_id = shortlistedEmail
where user_id=email;
END$$
DELIMITER ;
```

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`love_3`(IN shortlistedEmail VARCHAR(2000),email
VARCHAR(255))
READS SQL DATA
DETERMINISTIC
SQL SECURITY INVOKER
COMMENT 'INSERT NEW EMAIL ADN ITS
SHORTLISTEDEMAIL'
BEGIN
INSERT INTO shortlist values(email,shortlistedEmail);
END$$
DELIMITER ;

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`like_1`(IN email VARCHAR(255))
READS SQL DATA
DETERMINISTIC
SQL SECURITY INVOKER
COMMENT 'GIVE INFO OF ALL EMAIL_ID LIKED BY GIVEN
EMAIL_ID'
BEGIN
select *
from likes
where user_id=email;
END$$
DELIMITER ;

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`like_2`(IN likedEmail VARCHAR(2000),email
VARCHAR(255))
```

```
READS SQL DATA
DETERMINISTIC
SQL SECURITY INVOKER
COMMENT 'UPDATE LIKED_ID AT EMAIL WITH
LIKEDEMAIL'
BEGIN
UPDATE likes SET liked_id = likedEmail where
user_id=email;
END$$
DELIMITER ;

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`like_3`(IN likedEmail VARCHAR(2000),email
VARCHAR(255))
READS SQL DATA
DETERMINISTIC
SQL SECURITY INVOKER
COMMENT 'INSERT NEW EMAIL ADN ITS LIKEDEMAIL'
BEGIN
INSERT INTO likes values(email,likedEmail);
END$$
DELIMITER ;

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`dashboard_like_1`(IN email VARCHAR(255))
READS SQL DATA
DETERMINISTIC
SQL SECURITY INVOKER
COMMENT 'GIVE INFO OF ALL EMAIL_ID LIKED BY GIVEN
EMAIL_ID'
BEGIN
```

```
select *
from likes
where user_id=email;
END$$
DELIMITER ;

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`dashboard_like_2`(IN likedEmail VARCHAR(2000),email
VARCHAR(255))
READS SQL DATA
DETERMINISTIC
SQL SECURITY INVOKER
COMMENT 'UPDATE LIKED_ID AT EMAIL WITH
LIKEDEMAIL'
BEGIN
UPDATE likes SET liked_id = likedEmail where
user_id=email;
END$$
DELIMITER ;

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`dashboard_like_3`(IN likedEmail VARCHAR(2000),email
VARCHAR(255))
READS SQL DATA
DETERMINISTIC
SQL SECURITY INVOKER
COMMENT 'INSERT NEW EMAIL ADN ITS LIKEDEMAIL'
BEGIN
INSERT INTO likes values(email,likedEmail);
END$$
DELIMITER ;
```

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`login`(IN email VARCHAR(255))
READS SQL DATA
DETERMINISTIC
SQL SECURITY INVOKER
COMMENT 'CHECK EXISTENCE OF LOGIN CREDENTIALS'
BEGIN
select *
from users
where email_id=email;
END$$
DELIMITER ;

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`register_1`(IN email VARCHAR(255))
READS SQL DATA
DETERMINISTIC
SQL SECURITY INVOKER
COMMENT 'CHECK EXISTENCE OF EMAIL'
BEGIN
select *
from users
where email_id=email;
END$$
DELIMITER ;

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`register_2`
(IN email_id  VARCHAR(255),pwd  VARCHAR(100),
fore_name  VARCHAR(255), last_name  VARCHAR(255),
```

```
 profile_name  VARCHAR(255), gender  VARCHAR(10),
descript  VARCHAR(500), date_of_birth DATE,
 religion  VARCHAR(25), mother_tongue  VARCHAR(25),
height INT(5), marital_status  VARCHAR(25),
 private_id TINYINT(2), qualification  VARCHAR(255),
institution VARCHAR(255), occupation  VARCHAR(255),
country  VARCHAR(255), salary INT(12))
READS SQL DATA
DETERMINISTIC
SQL SECURITY INVOKER
COMMENT 'ADD NEW DATA TO USERS'
BEGIN
INSERT INTO users VALUES(email_id,pwd, fore_name,
last_name, profile_name, gender,
                                        descript, date_of_birth,
religion, mother_tongue, height,
                marital_status, private_id, qualification,
institution, occupation,
                country, salary);
END$$
DELIMITER ;

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`dashboard_userProfile_1`(IN email VARCHAR(255))
READS SQL DATA
DETERMINISTIC
SQL SECURITY INVOKER
COMMENT 'GIVE ALL INFO OF THE INPUT EMAIL'
BEGIN
select *
from users
where email_id=email;
```

```
END$$
DELIMITER ;

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`dashboard_userProfile_2`(IN email VARCHAR(255))
READS SQL DATA
DETERMINISTIC
SQL SECURITY INVOKER
COMMENT 'GIVE ALL EMAIL_ID LIKED BY GIVEN
EMAIL_ID'
BEGIN
select *
from likes
where user_id=email;
END$$
DELIMITER ;

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`dashboard_friendname_1`(IN email
VARCHAR(255),friendEmail VARCHAR(255))
READS SQL DATA
DETERMINISTIC
SQL SECURITY INVOKER
COMMENT 'GIVE CHATS OF ENTERED EMAIL AND
FRIENDEMAIL'
BEGIN
SELECT *
FROM chats
WHERE (user_id = email AND friend_id = friendEmail) OR
(user_id = friendEmail AND friend_id = email);
END$$
```

```
DELIMITER ;

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`dashboard_friendname_2`(IN email VARCHAR(255))
READS SQL DATA
DETERMINISTIC
SQL SECURITY INVOKER
COMMENT 'GIVE ALL EMAIL_ID LIKED BY GIVEN
EMAIL_ID'
BEGIN
select liked_id
from likes
where user_id=email;
END$$
DELIMITER ;

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`dashboard_friendname_3`(IN email VARCHAR(255))
READS SQL DATA
DETERMINISTIC
SQL SECURITY INVOKER
COMMENT 'CHECK EXISTENCE OF LOGIN CREDENTIALS'
BEGIN
select *
from users
where email_id=email;
END$$
DELIMITER ;
```

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`dashboard_chatlog`(IN email VARCHAR(255),friendEmail
VARCHAR(255), userChat VARCHAR(255))
READS SQL DATA
DETERMINISTIC
SQL SECURITY INVOKER
COMMENT 'INSERT ENTRY IN CHATS'
BEGIN
INSERT INTO chats VALUES(email,friendEmail,userChat);
END$$
DELIMITER ;

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`feedback`(IN email VARCHAR(255),feed TINYINT(2))
READS SQL DATA
DETERMINISTIC
SQL SECURITY INVOKER
COMMENT 'UPDATE FEEDBACK'
BEGIN
UPDATE users SET feedback = feed  WHERE email_id =
email;
END$$
DELIMITER ;

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`dpUpload_1`(IN email VARCHAR(255),imageName
VARCHAR(255))
READS SQL DATA
DETERMINISTIC
SQL SECURITY INVOKER
```

```
COMMENT 'UPDATE IMAGE'
BEGIN
UPDATE users SET image=imageName WHERE
email_id=email;
END$$
DELIMITER ;

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`dpUpload_2`()
READS SQL DATA
DETERMINISTIC
SQL SECURITY INVOKER
COMMENT 'GIVE EMAIL_ID AND ITS IMAGE'
BEGIN
SELECT email_id,image FROM users;
END$$
DELIMITER ;

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`method_1`(IN email VARCHAR(255))
READS SQL DATA
DETERMINISTIC
SQL SECURITY INVOKER
COMMENT 'GIVE ALL INFO FROM RLIKES TABLE
CORRESPONDING TO GIVEN EMAIL'
BEGIN
SELECT * FROM rlikes where user_id = email;
END$$
DELIMITER ;
```

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`method_2`(IN email VARCHAR(255), likedByEmail
VARCHAR(255))
READS SQL DATA
DETERMINISTIC
SQL SECURITY INVOKER
COMMENT 'UPDATE THE INFO CORRESPONDING TO
GIVEN EMAIL'
BEGIN
UPDATE rlikes SET liked_by_id = likedByEmail where
user_id= email;
END$$
DELIMITER ;

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE
`method_3`(IN email VARCHAR(255), likedByEmail
VARCHAR(255))
READS SQL DATA
DETERMINISTIC
SQL SECURITY INVOKER
COMMENT 'INSERT NEW DATA IN RLIKES'
BEGIN
INSERT INTO rlikes VALUES(user_id,liked_by_id);
END$$
DELIMITER ;
```