DMDD Final Project Simulation
Zixiao 20200328

Simulation

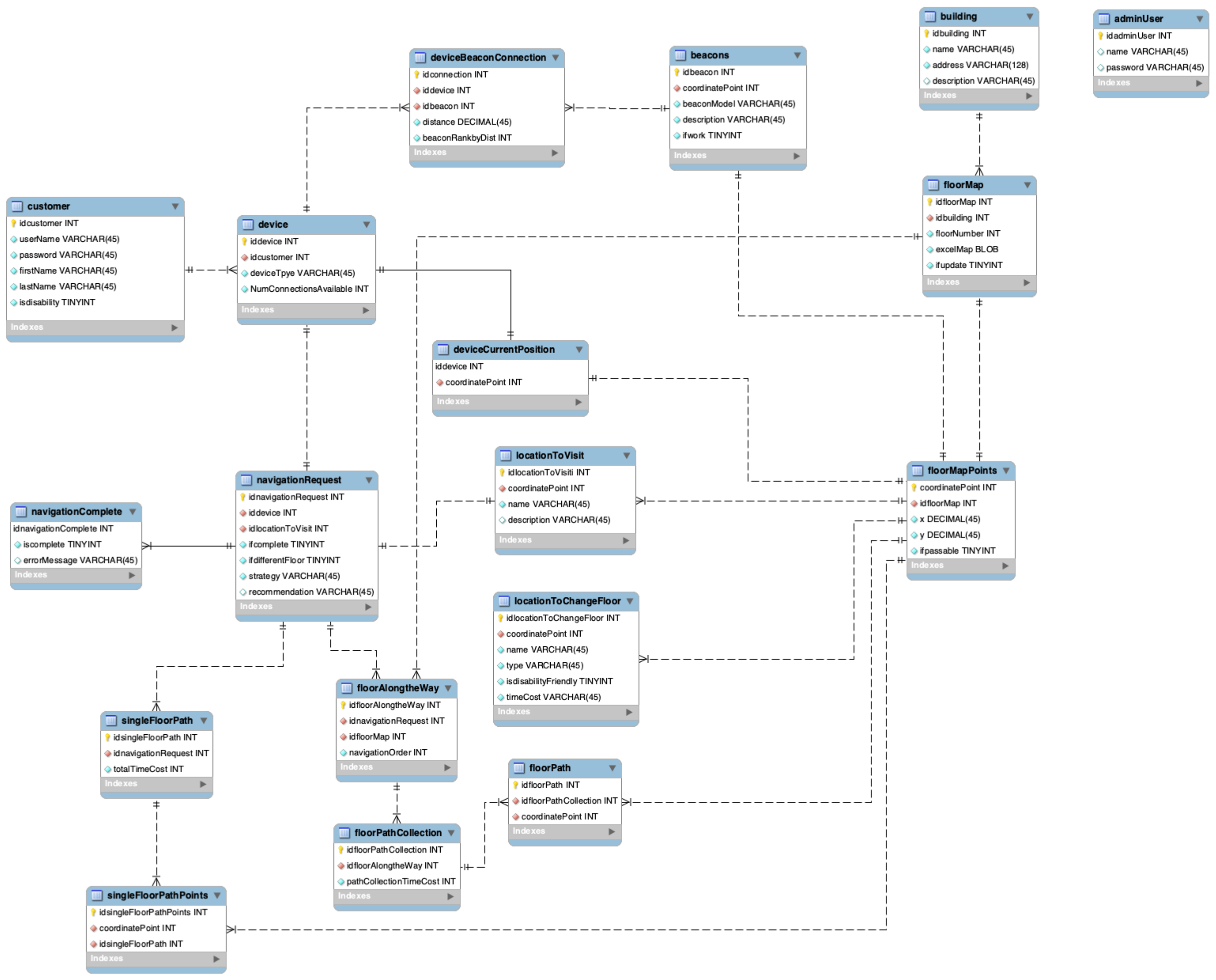
Content

Abstract

In this simulation, we didn't consider multithreading, which means we assume there is only one cumstomer is using the application.

So the main goal for this simulatio is to test if the database can store the data and interact with each other correctly.

The database for this simulation is as following:



This simulation include following things:

- 1. Test if the database can store right data into tables:
  - Map data simulation
    - `building`
    - `floorMap`
    - `floorMapPoints`
    - `locationToVisit`
    - `locationToChangeFloor`

        - `beacons`
    - User information simulation
        - `adminUser`
        - `customer`
- 2. Test if the connections between device and beacon are generated correctly
    - `device`
    - `deviceBeaconConnection`
- 3. Test if the positioning is updated correctly
    - `deviceCurrentPosition`
- 4. Test if the navigation is generated correctly
    - `navigationRequest`
- 5. Test if the paths are generated correctly
    - `singleFloorPath`
    - `singleFloorPathPoints`
    - `floorAlongtheWay`
    - `floorPathCollection`
    - `floorPath`
- 6. Test if the system is update correctly when navigation request is over
    - `navigationComplete`
    - `navigationRequest`

Connect to database

```
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
# import the magic code for using sql in jupyter notebook
%load_ext sql
# local database
# %sql mysql+pymysql://root:fjwwzx970814@localhost/mydb
# remote database
%sql mysql+pymysql://brickea_mac:fjwWZX970814@rm-0xih4pk94w41k3c5j8o.mysql.rds.aliyuncs.com/mydb
```

```
The sql extension is already loaded. To reload it, use:
  %reload_ext sql




'Connected: brickea_mac@mydb'
```

```
%%sql
show tables
```

```
 * mysql+pymysql://brickea_mac:***@rm-0xih4pk94w41k3c5j8o.mysql.rds.aliyuncs.com/mydb
   mysql+pymysql://root:***@localhost/mydb
18 rows affected.
```

Tables_in_mydb
adminuser
beacons
building
customer

device
devicebeaconconnection
devicecurrentposition
flooralongtheway
floormap
floormappoints
floorpath
floorpathcollection
locationtochangefloor
locationtovisit
navigationcomplete
navigationrequest
singlefloorpath
singlefloorpathpoints

```
import sqlalchemy as sqlManager
```

```
# Create connection with database
connection = sqlManager.create_engine('mysql+pymysql://brickea_mac:fjwWZX970814@rm-
0xih4pk94w41k3c5j8o.mysql.rds.aliyuncs.com/mydb?charset=utf8')
```

Map and user data simulation and test

Map data simulation

Because I only got one map sample as following

So I will use it as a template to generate 5 different floor maps for 2 different buildings

And I assume the coordinate original in the map is at its top left

The map is 20 X 21

Here I assume

- points within green line is a wall
- points within red circle is an elevator
- points within purple circle is a stair
- elevator and stair are in the same position for every floor
- elevator and stair can help human move to any floor

building - table test

```
# Generate building data
building = pd.DataFrame([
    [1,'building_1','address_1','description'],
    [2,'building_2','address_2','description'],
],columns=['idbuilding','name','address','description'])
building
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	idbuilding	name	address	description
0	1	building_1	address_1	description
1	2	building_2	address_2	description

```
# Insert data into database
pd.io.sql.to_sql(building,'building',connection,schema='mydb',if_exists='append',index=False)
```

```
%%sql
select * from building
```

```
 * mysql+pymysql://brickea_mac:***@rm-0xih4pk94w41k3c5j8o.mysql.rds.aliyuncs.com/mydb
   mysql+pymysql://root:***@localhost/mydb
2 rows affected.
```

idbuilding	name	address	description
1	building_1	address_1	description
2	building_2	address_2	description

floorMap - table test

```
def generate_floor_info(floor_number=[]):
    result = []
    floor_id = 0
    for building in range(len(floor_number)):
        for floor in range(floor_number[building]):
            result.append([floor_id+1,building+1,floor+1,
('building_'+str(building+1)+'_floor_'+str(floor+1)),True])
            floor_id+=1
    return result
```

```
floor_data = generate_floor_info(floor_number=[3,2])
floor_data
```

```
[[1, 1, 1, 'building_1_floor_1', True],
 [2, 1, 2, 'building_1_floor_2', True],
 [3, 1, 3, 'building_1_floor_3', True],
 [4, 2, 1, 'building_2_floor_1', True],
 [5, 2, 2, 'building_2_floor_2', True]]
```

```
floor_columns = ['idfloorMap','idbuilding','floorNumber','excelMap','ifupdate']
```

```
# Generate floor data for each building
floor = pd.DataFrame(data=floor_data,columns=floor_columns)
floor
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	idfloorMap	idbuilding	floorNumber	excelMap	ifupdate
0	1	1	1	building_1_floor_1	True
1	2	1	2	building_1_floor_2	True
2	3	1	3	building_1_floor_3	True
3	4	2	1	building_2_floor_1	True
4	5	2	2	building_2_floor_2	True

```
# Insert data into database
pd.io.sql.to_sql(floor,'floorMap',connection,schema='mydb',if_exists='append',index=False)
```

```
%%sql
select * from floorMap
```

```
 * mysql+pymysql://brickea_mac:***@rm-0xih4pk94w41k3c5j8o.mysql.rds.aliyuncs.com/mydb
   mysql+pymysql://root:***@localhost/mydb
5 rows affected.
```

idfloorMap	idbuilding	floorNumber	excelMap	ifupdate
1	1	1	b'building_1_floor_1'	1
2	1	2	b'building_1_floor_2'	1
3	1	3	b'building_1_floor_3'	1
4	2	1	b'building_2_floor_1'	1
5	2	2	b'building_2_floor_2'	1

floorMapPoints - table test

Here I assume bottom left is the coordinate original point

```
np.zeros((3,3))
```

```
array([[0., 0., 0.],
       [0., 0., 0.],
       [0., 0., 0.]])
```

```
def generate_floor_basic_points(x_len,y_len):
    return np.zeros((x_len,y_len))
```

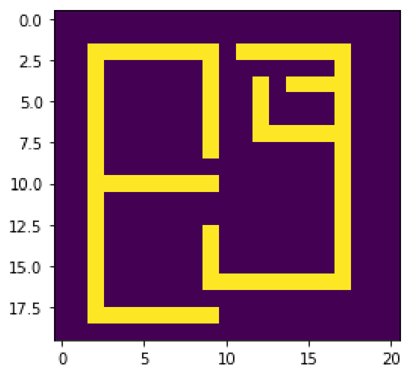
```
# The map is 20 X 21 (unit:meter)
floor_map_data = generate_floor_basic_points(20,21)
```

```
def generate_wall(floor_map,wall=[]):
    # wall should input coordinate of start point and end point
    # wall = [(start_x,start_y),(end_x,end_y)]
    start = wall[0]
    end = wall[1]
    for list_x in range(start[1],end[1]+1):
        for list_y in range(start[0],end[0]+1):
            floor_map[list_x][list_y] = 1
    return floor_map
```

```
# Generate wall
floor_map_data = generate_wall(floor_map_data,wall = [(2,2),(9,2)])
floor_map_data = generate_wall(floor_map_data,wall = [(9,2),(9,8)])
floor_map_data = generate_wall(floor_map_data,wall = [(2,2),(2,18)])
floor_map_data = generate_wall(floor_map_data,wall = [(2,18),(9,18)])
floor_map_data = generate_wall(floor_map_data,wall = [(2,10),(9,10)])
floor_map_data = generate_wall(floor_map_data,wall = [(9,13),(9,16)])
floor_map_data = generate_wall(floor_map_data,wall = [(9,16),(17,16)])
floor_map_data = generate_wall(floor_map_data,wall = [(11,2),(17,2)])
floor_map_data = generate_wall(floor_map_data,wall = [(17,2),(17,16)])
floor_map_data = generate_wall(floor_map_data,wall = [(14,4),(17,4)])
floor_map_data = generate_wall(floor_map_data,wall = [(12,4),(12,7)])
floor_map_data = generate_wall(floor_map_data,wall = [(12,7),(17,7)])
```

```
plt.imshow(floor_map_data)
```

```
<matplotlib.image.AxesImage at 0x2ae4576a208>
```



```
def generate_floor_map_point(floor_map,floor_id_list):
    result = []
    id_coordinate = 1
    for floor_id in floor_id_list:
        for y, y_values in enumerate(floor_map):
            for x, value in enumerate(y_values):
                passable = True if value == 0 else False
                result.append([id_coordinate,floor_id,x,y,passable])
                id_coordinate+=1
    return result
```

```
floor_map_points_data = generate_floor_map_point(floor_map_data,[1,2,3,4,5])
floor_map_points_columns = ['coordinatePoint','idfloorMap','x','y','ifpassable']
```

```
len(floor_map_points_data)
```

```
2100
```

```
20*21*5
```

```
2100
```

```
floor_map_points = pd.DataFrame(data = floor_map_points_data,columns=floor_map_points_columns)
```

```
# Insert data into database
pd.io.sql.to_sql(floor_map_points,'floorMapPoints',connection,schema='mydb',if_exists='append',index=False)
```

```
%%sql
select * from floorMapPoints
```

```
 * mysql+pymysql://brickea_mac:***@rm-0xih4pk94w41k3c5j8o.mysql.rds.aliyuncs.com/mydb
   mysql+pymysql://root:***@localhost/mydb
2100 rows affected.
```

coordinatePoint	idfloorMap	x	y	ifpassable
1	1	0	0	1
2	1	1	0	1
3	1	2	0	1
4	1	3	0	1
5	1	4	0	1
6	1	5	0	1
7	1	6	0	1
8	1	7	0	1
9	1	8	0	1
10	1	9	0	1
11	1	10	0	1
12	1	11	0	1
13	1	12	0	1

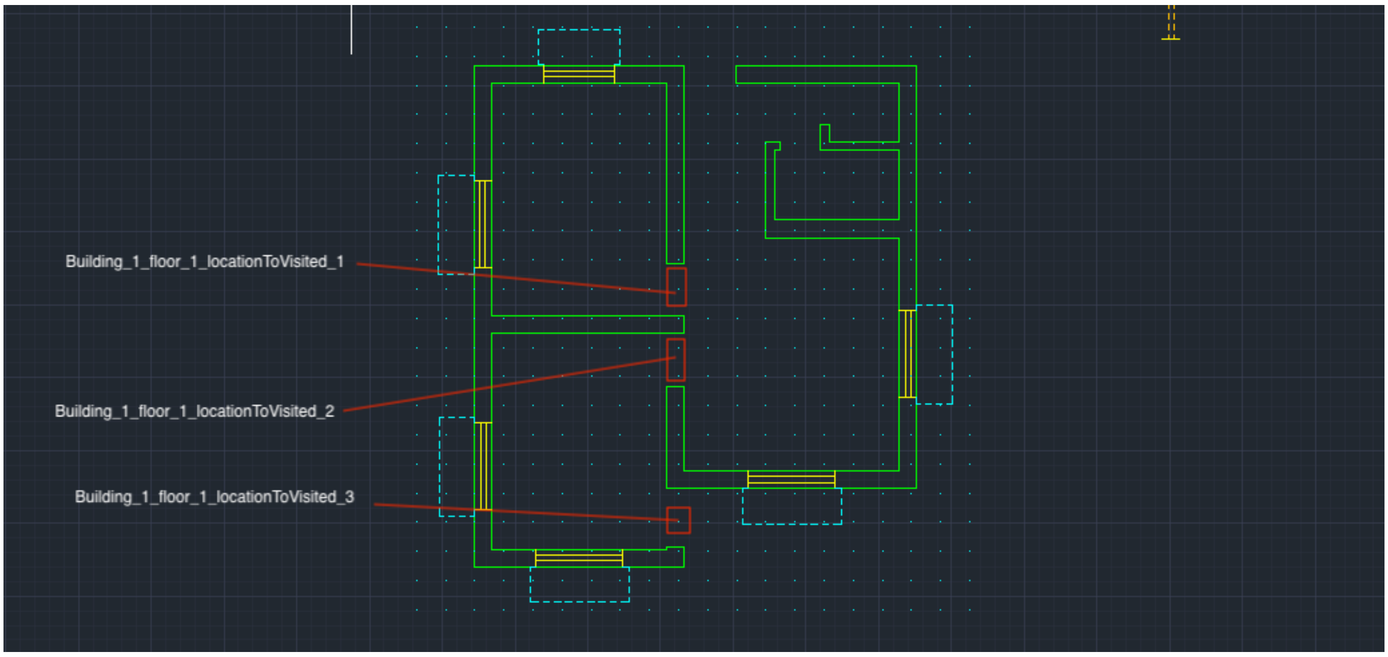
14	1	13	0	1
15	1	14	0	1
16	1	15	0	1
17	1	16	0	1
18	1	17	0	1
19	1	18	0	1
20	1	19	0	1
21	1	20	0	1
22	1	0	1	1
23	1	1	1	1
24	1	2	1	1
25	1	3	1	1
26	1	4	1	1
27	1	5	1	1
28	1	6	1	1
29	1	7	1	1
30	1	8	1	1
31	1	9	1	1
32	1	10	1	1
33	1	11	1	1
34	1	12	1	1
35	1	13	1	1
36	1	14	1	1
37	1	15	1	1
...				
2093	5	13	19	1
2094	5	14	19	1
2095	5	15	19	1
2096	5	16	19	1
2097	5	17	19	1
2098	5	18	19	1
2099	5	19	19	1
2100	5	20	19	1

locationToVisit - table test

The location in building 1 floor 1 is as follow

Coordinate of location 1 and location 2:

- location 1: (9,9)
- location 2: (9,11)
- location 3: (9,17)


Building_1_floor_1_locationToVisited_1
Building_1_floor_1_locationToVisited_2
Building_1_floor_1_locationToVisited_3

```
floor = 1
x = 9
y= 17
sql = 'select * from'\
' (select * from floorMapPoints fmp'\
' where fmp.idfloorMap = ' + str(floor) + ') fp'\
' where fp.x = ' + str(x) + ' and fp.y = '+ str(y)

print(sql)
df = pd.read_sql(sql,connection)['coordinatePoint'][0]
df
```

```
select * from (select * from floorMapPoints fmp where fmp.idfloorMap = 1) fp where fp.x = 9 and fp.y = 17




367
```

```
# Generate locations for building 1 floor 1
def generate_location_to_visited(floor=[], locations_coordinate=[]):
    result = []
    id_location = 1
    for floor_id in floor:
        for location_coordinate in locations_coordinate:
            floor = floor_id
            x = location_coordinate[0]
            y = location_coordinate[1]
            location_no = len(locations_coordinate) if id_location % len(locations_coordinate)==0 else
id_location % len(locations_coordinate)

            sql = 'select * from'\
            ' (select * from floorMapPoints fmp'\
            ' where fmp.idfloorMap = ' + str(floor) + ') fp'\
            ' where fp.x = ' + str(x) + ' and fp.y = '+ str(y)
            map_point_id = pd.read_sql(sql,connection)['coordinatePoint'][0]


result.append([id_location,map_point_id,'floor_'+str(floor)+'_location_'+str(location_no),'floor_'+str(floor)+'_
location_'+str(location_no)+'_description'])
```

```
            id_location+=1

    return result
```

```
location_to_visited_data = generate_location_to_visited(floor=[1,2,3,4,5],locations_coordinate=[(9,9),(9,11),
(9,17)])
```

```
location_to_visited_columns = ['idlocationToVisiti','coordinatePoint','name','description']
location_to_visited = pd.DataFrame(data=location_to_visited_data,columns=location_to_visited_columns)
```

```
pd.io.sql.to_sql(location_to_visited,'locationToVisited',connection,schema='mydb',if_exists='append',index=False
)
```

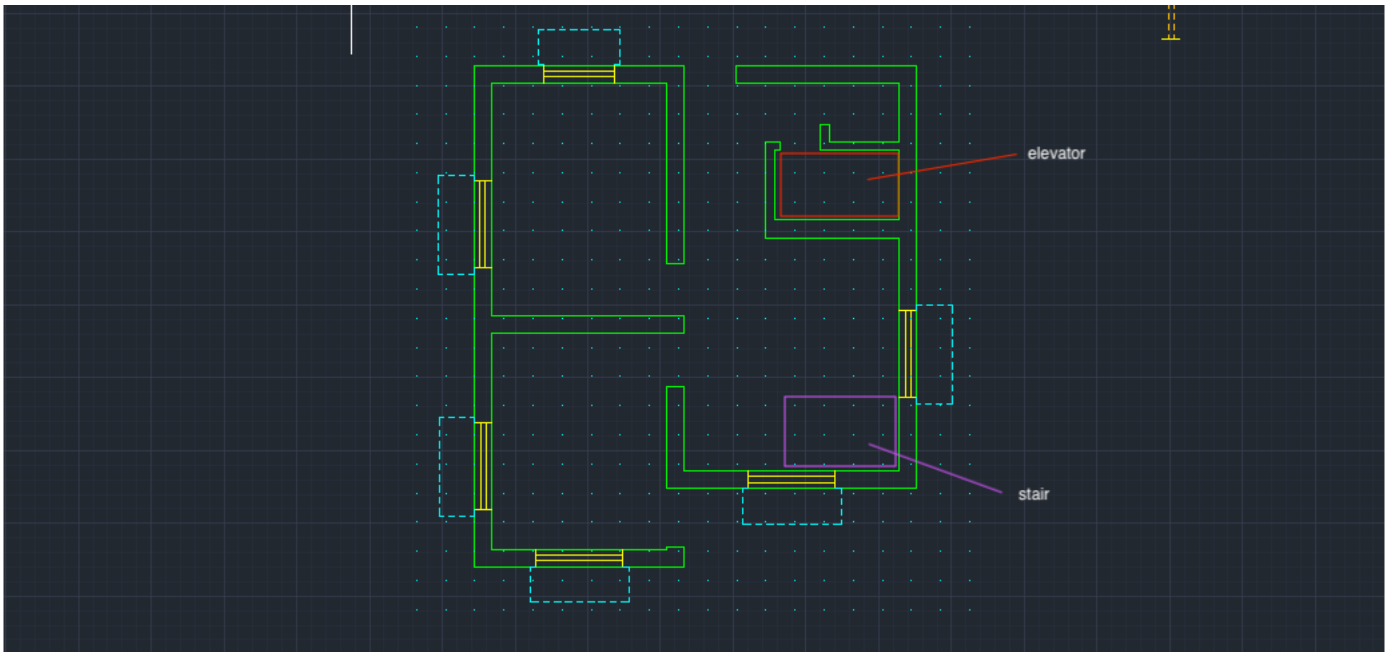
```
%%sql
select * from locationToVisited
```

```
 * mysql+pymysql://brickea_mac:***@rm-0xih4pk94w41k3c5j8o.mysql.rds.aliyuncs.com/mydb
   mysql+pymysql://root:***@localhost/mydb
15 rows affected.
```

idlocationToVisiti	coordinatePoint	name	description
1	199	floor_1_location_1	floor_1_location_1_description
2	241	floor_1_location_2	floor_1_location_2_description
3	367	floor_1_location_3	floor_1_location_3_description
4	619	floor_2_location_1	floor_2_location_1_description
5	661	floor_2_location_2	floor_2_location_2_description
6	787	floor_2_location_3	floor_2_location_3_description
7	1039	floor_3_location_1	floor_3_location_1_description
8	1081	floor_3_location_2	floor_3_location_2_description
9	1207	floor_3_location_3	floor_3_location_3_description
10	1459	floor_4_location_1	floor_4_location_1_description
11	1501	floor_4_location_2	floor_4_location_2_description
12	1627	floor_4_location_3	floor_4_location_3_description
13	1879	floor_5_location_1	floor_5_location_1_description
14	1921	floor_5_location_2	floor_5_location_2_description
15	2047	floor_5_location_3	floor_5_location_3_description

locationToChangeFloor - table test

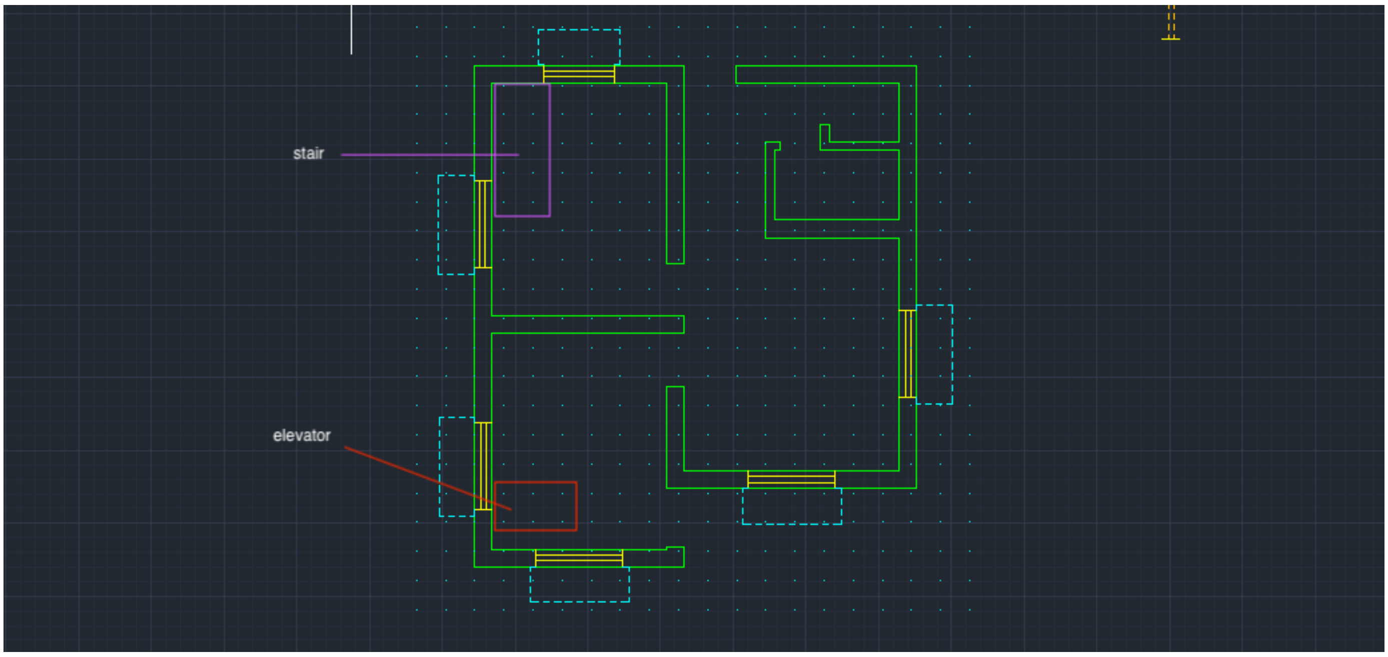
Building 1 floor 1 2 3



In building 1:

- elevator: (13,4)
- start: (13,14)

Building 2 floor 1 2



In building 2:

- elevator: (4,16)
- stair: (4,4)

```
# Generate locationToChangeFloor for building 1 floor 1
def generate_location_to_change_floor(floor=[], name=[], locations_coordinate=[]):
    result = []
    id_location = 1
    for floor_id in floor:
        for location_coordinate in locations_coordinate:
            floor = floor_id
            x = location_coordinate[0]
            y = location_coordinate[1]
            location_no = len(locations_coordinate) if id_location % len(locations_coordinate)==0 else
id_location % len(locations_coordinate)
```

```
            location_type = name[location_no-1]
            is_friendly = True if name[location_no-1] == 'elevator' else False
            timeCost = 5 if is_friendly else 10

            sql = 'select * from'\
            ' (select * from floorMapPoints fmp'\
            ' where fmp.idfloorMap = ' + str(floor) + ') fp'\
            ' where fp.x = ' + str(x) + ' and fp.y = '+ str(y)
            map_point_id = pd.read_sql(sql,connection)['coordinatePoint'][0]

            result.append([id_location,
                           map_point_id,

'floor_'+str(floor)+'_location_to_change_floor_'+str(location_no)+'_'+name[location_no-1],
                           location_type,
                           is_friendly,
                           timeCost])
            id_location+=1

    return result
```

```
location_to_change_floor_data = generate_location_to_change_floor(floor=[1,2,3],name=['elevator','stair'],
locations_coordinate=[(13,4),(13,14)])
```

```
location_to_change_floor_columns =
['idlocationToChangeFloor','coordinatePoint','name','type','isdisabilityFriendly','timeCost']
location_to_change_floor =
pd.DataFrame(data=location_to_change_floor_data,columns=location_to_change_floor_columns)
location_to_change_floor
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	idlocationToChangeFloor	coordinatePoint	name	type	isdisabilityFriendly	timeCost
0	1	98	floor_1_location_to_change_floor_1_elevator	elevator	True	5
1	2	308	floor_1_location_to_change_floor_2_stair	stair	False	10
2	3	518	floor_2_location_to_change_floor_1_elevator	elevator	True	5
3	4	728	floor_2_location_to_change_floor_2_stair	stair	False	10
4	5	938	floor_3_location_to_change_floor_1_elevator	elevator	True	5
5	6	1148	floor_3_location_to_change_floor_2_stair	stair	False	10

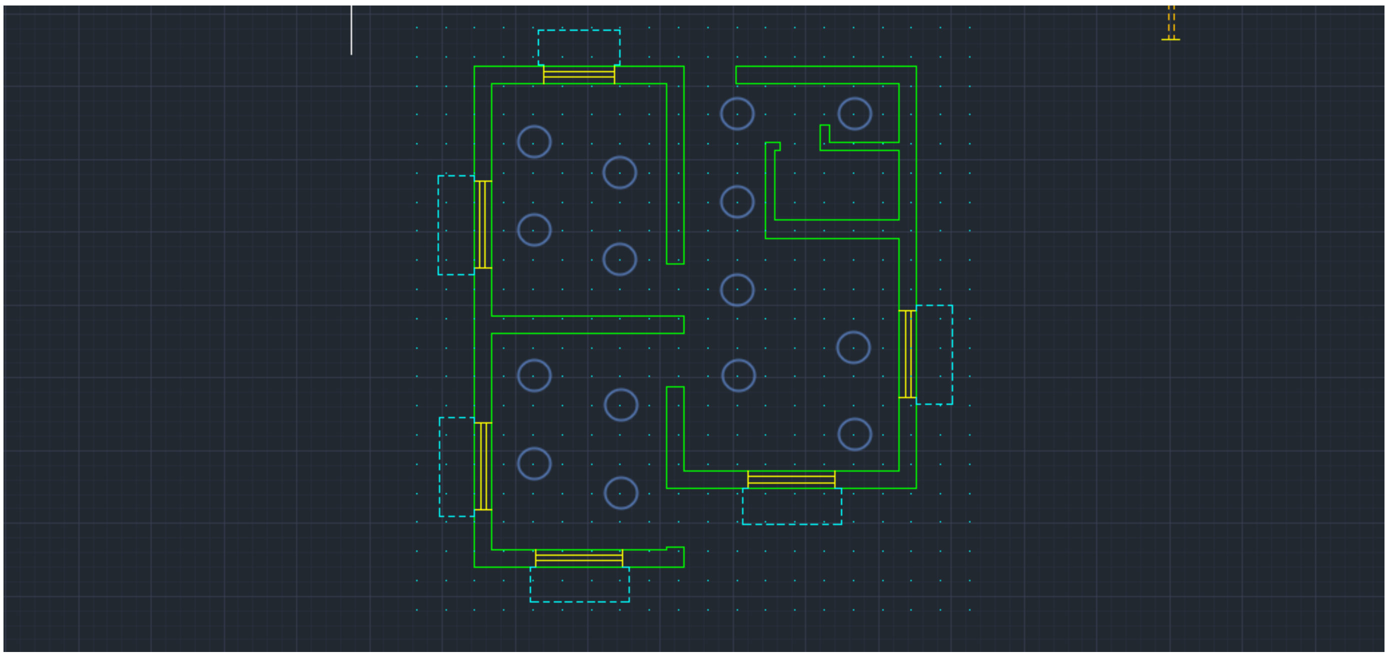
```
pd.io.sql.to_sql(location_to_change_floor,'locationToChangeFloor',connection,schema='mydb',if_exists='append',in
dex=False)
```

```
%%sql
select * from locationToChangeFloor
```

```
 * mysql+pymysql://brickea_mac:***@rm-0xih4pk94w41k3c5j8o.mysql.rds.aliyuncs.com/mydb
   mysql+pymysql://root:***@localhost/mydb
6 rows affected.
```

idlocationToChangeFloor	coordinatePoint	name	type	isdisabilityFriendly	timeCost
1	98	floor_1_location_to_change_floor_1_elevator	elevator	1	5
2	308	floor_1_location_to_change_floor_2_stair	stair	0	10
3	518	floor_2_location_to_change_floor_1_elevator	elevator	1	5
4	728	floor_2_location_to_change_floor_2_stair	stair	0	10
5	938	floor_3_location_to_change_floor_1_elevator	elevator	1	5
6	1148	floor_3_location_to_change_floor_2_stair	stair	0	10

beacons - table test



These are beacons positions inside floors

I assume all floors use the same pattern

The position:

- (4,4)
- (7,5)
- (4,7)
- (7,8)
- (11,3)
- (15,3)
- (11,6)
- (11,9)
- (11,12)
- (15,11)
- (15,14)

```python
# Generate beacon for all floor
def generate_beacon_location(floor=[], beacons_locations_coordinate=[]):
    result = []
    id_location = 1
    for floor_id in floor:
        for location_coordinate in beacons_locations_coordinate:
            floor = floor_id
            x = location_coordinate[0]
            y = location_coordinate[1]
            location_no = len(beacons_locations_coordinate) if id_location %
len(beacons_locations_coordinate)==0 else id_location % len(beacons_locations_coordinate)

            sql = 'select * from'\
            ' (select * from floorMapPoints fmp'\
            ' where fmp.idfloorMap = ' + str(floor) + ') fp'\
            ' where fp.x = ' + str(x) + ' and fp.y = '+ str(y)
            map_point_id = pd.read_sql(sql,connection)['coordinatePoint'][0]

            result.append([id_location,
                           map_point_id,
                           'floor_'+str(floor)+'_beacon_model_'+str(location_no),
                           'floor_'+str(floor)+'_beacon_'+str(location_no)+'_description',
                           True])
            id_location+=1

    return result
```

```
beacon_location_data = generate_beacon_location(floor=[1,2,3,4,5],beacons_locations_coordinate=[(4,4),(7,5),
(4,7),(7,8),(11,3),(15,3),(11,6),(11,9),(11,12),(15,11),(15,14)])
```

```
beacon_location_columns = ['idbeacon','coordinatePoint','beaconModel','description','ifwork']
beacon_location = pd.DataFrame(data=beacon_location_data,columns=beacon_location_columns)
```

```
pd.io.sql.to_sql(beacon_location,'beacons',connection,schema='mydb',if_exists='append',index=False)
```

```
%%sql
select * from beacons
```

```
 * mysql+pymysql://brickea_mac:***@rm-0xih4pk94w41k3c5j8o.mysql.rds.aliyuncs.com/mydb
   mysql+pymysql://root:***@localhost/mydb
55 rows affected.
```

idbeacon	coordinatePoint	beaconModel	description	ifwork
1	89	floor_1_beacon_model_1	floor_1_beacon_1_description	1
2	113	floor_1_beacon_model_2	floor_1_beacon_2_description	1
3	152	floor_1_beacon_model_3	floor_1_beacon_3_description	1
4	176	floor_1_beacon_model_4	floor_1_beacon_4_description	1
5	75	floor_1_beacon_model_5	floor_1_beacon_5_description	1
6	79	floor_1_beacon_model_6	floor_1_beacon_6_description	1
7	138	floor_1_beacon_model_7	floor_1_beacon_7_description	1
8	201	floor_1_beacon_model_8	floor_1_beacon_8_description	1
9	264	floor_1_beacon_model_9	floor_1_beacon_9_description	1
10	247	floor_1_beacon_model_10	floor_1_beacon_10_description	1
11	310	floor_1_beacon_model_11	floor_1_beacon_11_description	1
12	509	floor_2_beacon_model_1	floor_2_beacon_1_description	1

13	533	floor_2_beacon_model_2	floor_2_beacon_2_description	1
14	572	floor_2_beacon_model_3	floor_2_beacon_3_description	1
15	596	floor_2_beacon_model_4	floor_2_beacon_4_description	1
16	495	floor_2_beacon_model_5	floor_2_beacon_5_description	1
17	499	floor_2_beacon_model_6	floor_2_beacon_6_description	1
18	558	floor_2_beacon_model_7	floor_2_beacon_7_description	1
19	621	floor_2_beacon_model_8	floor_2_beacon_8_description	1
20	684	floor_2_beacon_model_9	floor_2_beacon_9_description	1
21	667	floor_2_beacon_model_10	floor_2_beacon_10_description	1
22	730	floor_2_beacon_model_11	floor_2_beacon_11_description	1
23	929	floor_3_beacon_model_1	floor_3_beacon_1_description	1
24	953	floor_3_beacon_model_2	floor_3_beacon_2_description	1
25	992	floor_3_beacon_model_3	floor_3_beacon_3_description	1
26	1016	floor_3_beacon_model_4	floor_3_beacon_4_description	1
27	915	floor_3_beacon_model_5	floor_3_beacon_5_description	1
28	919	floor_3_beacon_model_6	floor_3_beacon_6_description	1
29	978	floor_3_beacon_model_7	floor_3_beacon_7_description	1
30	1041	floor_3_beacon_model_8	floor_3_beacon_8_description	1
31	1104	floor_3_beacon_model_9	floor_3_beacon_9_description	1
32	1087	floor_3_beacon_model_10	floor_3_beacon_10_description	1
33	1150	floor_3_beacon_model_11	floor_3_beacon_11_description	1
34	1349	floor_4_beacon_model_1	floor_4_beacon_1_description	1
35	1373	floor_4_beacon_model_2	floor_4_beacon_2_description	1
36	1412	floor_4_beacon_model_3	floor_4_beacon_3_description	1
37	1436	floor_4_beacon_model_4	floor_4_beacon_4_description	1
38	1335	floor_4_beacon_model_5	floor_4_beacon_5_description	1
39	1339	floor_4_beacon_model_6	floor_4_beacon_6_description	1
40	1398	floor_4_beacon_model_7	floor_4_beacon_7_description	1
41	1461	floor_4_beacon_model_8	floor_4_beacon_8_description	1
42	1524	floor_4_beacon_model_9	floor_4_beacon_9_description	1
43	1507	floor_4_beacon_model_10	floor_4_beacon_10_description	1
44	1570	floor_4_beacon_model_11	floor_4_beacon_11_description	1
45	1769	floor_5_beacon_model_1	floor_5_beacon_1_description	1
46	1793	floor_5_beacon_model_2	floor_5_beacon_2_description	1
47	1832	floor_5_beacon_model_3	floor_5_beacon_3_description	1
48	1856	floor_5_beacon_model_4	floor_5_beacon_4_description	1
49	1755	floor_5_beacon_model_5	floor_5_beacon_5_description	1
50	1759	floor_5_beacon_model_6	floor_5_beacon_6_description	1
51	1818	floor_5_beacon_model_7	floor_5_beacon_7_description	1
52	1881	floor_5_beacon_model_8	floor_5_beacon_8_description	1
53	1944	floor_5_beacon_model_9	floor_5_beacon_9_description	1

54	1927	floor_5_beacon_model_10	floor_5_beacon_10_description	1
55	1990	floor_5_beacon_model_11	floor_5_beacon_11_description	1

User information simulation

- adminUser
- customer

```
# Add admin user data
admin_user_data = [[1,'admin','admin']]
```

```
admin_user_columns = ['idadminUser','name','password']
admin_user = pd.DataFrame(data = admin_user_data,columns=admin_user_columns)
```

```
pd.io.sql.to_sql(admin_user,'adminUser',connection,schema='mydb',if_exists='append',index=False)
```

```
%%sql
select * from adminUser
```

```
 * mysql+pymysql://brickea_mac:***@rm-0xih4pk94w41k3c5j8o.mysql.rds.aliyuncs.com/mydb
   mysql+pymysql://root:***@localhost/mydb
1 rows affected.
```

idadminUser	name	password
1	admin	admin

```
# Add customer user data
customer_user_data = []
for i in range(10):
    customer_user_data.append([
        i+1,
        'user_name_'+str(i),
        'password_'+str(i),
        'first_name_'+str(i),
        'last_name_'+str(i),
        False
    ])
customer_user_data.append([
    11,
    'disability_1',
    'password_'+str(i),
    'first_name_'+str(i),
    'last_name_'+str(i),
    True
])
```

```
customer_user_columns = ['idcustomer','userName','password','firstName','lastName','isdisability']
customer_user = pd.DataFrame(data = customer_user_data,columns=customer_user_columns)
```

```
pd.io.sql.to_sql(customer_user,'customer',connection,schema='mydb',if_exists='append',index=False)
```

```
%%sql
select * from customer
```

```
 * mysql+pymysql://brickea_mac:***@rm-0xih4pk94w41k3c5j8o.mysql.rds.aliyuncs.com/mydb
   mysql+pymysql://root:***@localhost/mydb
11 rows affected.
```

idcustomer	userName	password	firstName	lastName	isdisability
1	user_name_0	password_0	first_name_0	last_name_0	0
2	user_name_1	password_1	first_name_1	last_name_1	0
3	user_name_2	password_2	first_name_2	last_name_2	0
4	user_name_3	password_3	first_name_3	last_name_3	0
5	user_name_4	password_4	first_name_4	last_name_4	0
6	user_name_5	password_5	first_name_5	last_name_5	0
7	user_name_6	password_6	first_name_6	last_name_6	0
8	user_name_7	password_7	first_name_7	last_name_7	0
9	user_name_8	password_8	first_name_8	last_name_8	0
10	user_name_9	password_9	first_name_9	last_name_9	0
11	disability_1	password_9	first_name_9	last_name_9	1