# Database design

## Tables

Room(rName, level, prevRoom, coords, description)

Cords refers the location of the item on the floor plan

Description is a general description of the room’s funcion

Staff(tutorID, lastName, firstName, room)

Room is the office for the member of staff

Route(from, to, route)

Route is the text description of the route

## Route finding algorithm

### Skeleton and descriptions:

**Route(from, to, stepFree): List of String**

1. Backtrack to height 0 node for both from and to storing list of visited nodes
2. A)If different level then add the lift or stairs to the backtrack at the correct end and return routeNav(btcFrom, null, LR) + “travel to level to.level + routeNav(btcTo, null, RL)
3. B)If same level then get earliest common node ‘ecn’
4. Return routeNav(btcFrom, ecn, LR) + routeNav(btcTo, ecn, RL)

**Backtrack(from): List of Room**

// using prevRoom value in table create an list of rooms backtracking from the parameter until no prevRoom found. This is known as a height 0 node.

**ECN(btcFrom, btcTo): Room**

// work from the last item in the backtracked lists finding out the leftmost room that the lists have in common.

**RouteNav(backtrack, ecn, direction): List of String directions**

// use backtrack list to call route table up to ecn. Direction is either left to right or right to left when using backtrack.

### Pseudo code

**Route(from, to, stepFree): List of String**

btcFrom = backtrack(from)

btcTo = backtrack(to)

if(from.level == to.level) then

ecn = ECN(btcFrom, btcTo)

return routeNav(from, ecn, LR) + routeNav(ecn, to, RL)

end if

else if( stepFree = true ) then

**Backtrack(from): List of Room**

List btc

btc.add(from)

while (prevRoom = *SELECT prevRoom FROM Room WHERE rName = from.rName* != ‘’) do

btc.addprevRoom

od

**ECN(btcFrom, btcTo): Room**

fromPoint = btcFrom.length – 1

toPoint = btcTo.length -1

while( btcFrom.get(fromPointer) = btcFrom.get(toPointer) ) do

fromPointer = fromPointer -1

toPointer = toPointer – 1

od

return btcFrom.get(fromPointer + 1)

**RouteNav(backtrack, ecn, direction): List of String**

If direction = “LR”

pointer = 0

change = 1

Else if direction = “RL”

pointer = backtrack.length – 1

change = -1

While( backtrack.get(pointer) != ecn && pointer >=0 && pointer < backtrack.length ) do

route.add( SELECT directions from Route WHERE from = backtrack.get(pointer) AND to = backtrack.get(pointer + change) )

pointer = pointer + change

od

## Searching for a Tutor, Pseudo code

Search bar when pressed will contain a hint “Search”

Create a method which retrieves a specific field (e.g. names) from the database. This method will be called within the setup of the search bar. Use this information as a parameter for the setLastSuggestions within the search bar. E.g.

materialSearchBar.setLastSuggestions(suggestList);

Add a text change listener to the setup of the search bar. This should contain override methods (beforeTextChanged),(onTextChanged)and (afterTextChanged).

Also, create a set on search action listener. This will have a methods such as a recyclerview adapter, start search when the search has been confirmed etc.

The startSearch method will contain an adapter which will adapt the information from the database to a recycler view.

The default search result should be set to all results, this will show users all the available options when they first use the search feature.