PLAN FOR PROTOTYPE

Building the system:

- Using java to program: retrieve relevant data from an existing database and form a nested hashmap; then based on the data retrieved, perform calculations for each tenant's upper and lower thresholds (for giving suggestions regarding what time to do housework and use communal spaces for that person).

- Compare the average presence count for each person at each hour over the week with the calculated upper and lower bound. If the person's hourly presence count > upper bound, give housework suggestions. If count > upper bound && average presence count for all tenants at that hour < lower bound, give activities suggestions (for using shared facilities etc).

- Send suggestions on standard output
- (ideally, the system should send all suggestions in the form of a schedule/ timetable for each person at the start of each week - at which the program is run, and send reminders shortly before each scheduled event occurs).

- Testing for each section and integrated system (as included in waterfall model)
- To make it more reasonable, we are only considering the day_hour *from 8a.m. to 22p.m.*; so that people won't be suggested to do housework etc at the middle of the night. So our calculations and the data that need to be fetched via Hashmap will need to be adjusted accordingly.

To simulate the existing SSH products:

- hard code the SSH Cloud database

(Sql is already existing and is able to know whether someone has entered or not)

Group plan:

- Build a github repository on friday (Research on github)
- Saabarin -> hashmap
- Jasmine -> calculations
- Asmaa -> make a database using sql and comparison

Summary of what to do for everyone before Friday:

1. Understand your role
2. Take a look at git / workflow if you have time
3. If you want, start working on your part.

Workflow (from what we learnt in Software Engineering):

Waterfall model:

1. Requirements analysis:

Must have: 1) suggest the time to do housework / to use shared facilities for each user 2) perform calculations for thresholds and comparisons based on fetched data; 3) develop with “accessibility, availability, performance, security, usability” in mind

Should have: 1) a more beautiful & organised output display (e.g. a nicely-printed timetable-like result in standard output)

Could have: 1) Add accessibility restrictions: so that only tenants from the accommodation is able to access the result - for example, require a password 2) Caching the hashmap & thresholds & comparison results & timetable;

Won’t have: any follow-ups on how well the suggestions are carried out; more data collection and analysis on more detailed occupancy information (e.g. when and where are each tenant within the flat)

2. Design:
    1) SQL will be used for the creation of our database.
    2) SQL aImplemented into our program through the use of JDBC
    3) Values will be extracted from database and placed into a nested hashmap
    4) Hashmap will then be used for calculations
    5) Then use of if statement for student who wants to find out result of two different suggestions:
        a. When they should do certain chores
        b. What time they can use certain facilities in the shared accommodation

3. Implementation & unit testing:
    1) Will be implemented using Java
    2) Changes will be add to the github repo
    3) Once each function is done it will be tested by all team members individually and a s a group and we will discuss
    4)
4. Integration & system testing:
    1)Implementing unit testing using jUnit i
    2)Testing the whole system and seeing if it works (e.g. by running Main.java)

5. Operation & maintenance:
    1) As it is a prototype , it will not be deployed but future product will be installed in SSH software.

Work links:

Establishing JDBC Connection in Java - GeeksforGeeks - this is for connection to SQL via Java

The Ultimate Github Collaboration Guide | by Jonathan Mines | Medium - this is about how to work collaboratively on Git

Git and GitHub for Beginners - Crash Course - this is a very good git and github course, worth watching: to be noted that you need to replace ‘master’ by ‘main’ for all commands mentioned in this video & better to use Git Bash for Windows & SSH keys info is not up-to-date, so better use the link down the video for latest information.

Bing Videos - this is a very straightforward and easy-to-follow video on how we should collaborate on GitHub (note that 'master' should be replaced by 'main' in the commands)

Common Git commands:
git clone (followed by clone URL) - to clone a repository from GitHub to your local machine
git status - to know the current status of git (check if everything is staged / committed)
git pull origin main - to update your local repository to the latest version on GitHub
git add . - to stage all changes you made locally and make things ready for commit
git commit -m "a message explaining the changes you made" - to commit (save) the staged changes, make them ready to be pushed (to a branch or to main)
git checkout -b branch-name - to create a new branch and switch to that branch
git checkout main - to switch to main
git push origin branch-name - to upload your local branch to GitHub, for other people to review, before changes are merged to main
git log - to view the history of a git repository

Example database (rough):

Week 1:

Time	Users\Day	Mon	Tue	Wed	Thu	Fri	Sat	Sun
0-1	A	1	1	1	1	1	1	1
	B	1	1	1	1	1	1	1
1-2	A	1	1	1	1	1	1	1
	B	1	1	1	1	1	1	1
2-3	A	1	1	1	1	1	1	1
	B	1	1	1	1	1	1	1
3-4	A	1	1	1	1	1	1	1
	B	1	1	1	1	1	1	1
4-5	A	1	1	1	1	1	1	1
	B	1	1	1	1	1	1	1
5-6	A	1	1	1	1	1	1	1
	B	1	1	1	1	1	1	1
6-7	A	1	1	1	1	1	1	1

	B	1	1	1	1	1	1	1
7-8	A	0	0	1	0	0	1	1
	B	1	1	1	1	1	1	1
8-9	A	0	0	0	0	0	1	1
	B	1	1	1	1	1	1	1
9-10	A	0	0	0	0	0	0	0
	B	1	1	1	1	1	1	1
10-11	A	0	0	0	0	0	0	0
	B	0	0	1	1	0	1	1
11-12	A	0	0	0	0	0	1	0
	B	0	0	0	0	0	1	1
12-13	A	0	0	0	0	0	0	0
	B	0	0	0	0	0	0	0
13-14	A	0	0	0	0	0	0	0
	B	0	1	0	0	0	0	0
14-15	A	0	0	0	0	0	0	0
	B	0	1	0	1	1	0	1
15-16	A	0	0	0	0	0	0	0
	B	1	1	0	1	1	0	1
16-17	A	1	0	1	0	0	1	0
	B	1	1	1	0	1	0	1
17-18	A	1	0	1	1	1	1	0
	B	1	0	0	0	0	0	1
18-19	A	0	1	0	0	0	1	0

	B	0	0	0	1	0	0	0
19-20	A	0	1	1	1	1	1	1
	B	1	1	0	0	0	1	0
20-21	A	1	1	1	1	1	1	1
	B	1	1	0	1	1	1	1
21-22	A	1	1	1	1	1	1	1
	B	1	1	1	1	1	1	1
22-23	A	1	1	1	1	1	1	1
	B	1	1	1	1	1	1	1
23-0	A	1	1	1	1	1	1	1
	B	1	1	1	1	1	1	1

Week 2:

Time	Users\Day	Mon	Tue	Wed	Thu	Fri	Sat	Sun
0-1	A	1	1	1	1	1	1	1
	B	1	1	1	1	1	1	1
1-2	A	1	1	1	1	1	1	1
	B	1	1	1	1	1	1	1
2-3	A	1	1	1	1	1	1	1
	B	1	1	1	1	1	1	1
3-4	A	1	1	1	1	1	1	1
	B	1	1	1	1	1	1	1
4-5	A	1	1	1	1	1	1	1
	B	1	1	1	1	1	1	1
5-6	A	1	1	1	1	1	1	1

	B	1	1	1	1	1	1	1
6-7	A	0	0	1	0	0	1	1
	B	1	1	1	1	1	1	1
7-8	A	1	1	0	1	1	0	0
	B	1	1	1	1	1	1	1
8-9	A	0	1	0	1	0	1	1
	B	1	1	1	1	1	1	1
9-10	A	0	0	0	0	0	1	0
	B	1	1	1	1	1	1	1
10-11	A	0	0	0	0	1	0	0
	B	0	0	0	1	0	1	1
11-12	A	0	0	0	0	0	0	0
	B	1	0	0	1	0	1	0
12-13	A	0	0	0	1	0	0	0
	B	1	0	0	0	0	1	0
13-14	A	0	1	0	1	0	0	1
	B	0	1	0	1	0	0	0
14-15	A	1	0	0	0	0	0	1
	B	0	1	0	0	0	0	1
15-16	A	0	0	1	0	1	0	0
	B	0	0	0	0	0	0	1
16-17	A	0	0	1	0	0	1	0
	B	1	0	0	0	0	0	0
17-18	A	0	1	1	0	0	1	0
	B	1	0	0	0	1	0	0

18-19	A	0	0	0	1	0	1	1
	B	1	1	0	1	0	1	0
19-20	A	1	1	1	1	1	1	1
	B	0	1	0	0	0	0	1
20-21	A	1	1	1	1	1	0	1
	B	1	1	0	1	0	0	1
21-22	A	1	1	1	1	1	0	1
	B	1	1	1	1	0	0	1
22-23	A	1	1	1	1	1	1	1
	B	1	1	1	1	0	0	1
23-0	A	1	1	1	1	1	1	1
	B	1	1	1	1	1	1	1

Week 3:

Time	Users\Day	Mon	Tue	Wed	Thu	Fri	Sat	Sun
0-1	A	1	1	1	1	1	1	1
	B	1	1	1	1	1	1	1
1-2	A	1	1	1	1	1	1	1
	B	1	1	1	1	1	1	1
2-3	A	1	1	1	1	1	1	1
	B	1	1	1	1	1	1	1
3-4	A	1	1	1	1	1	1	1
	B	1	1	1	1	1	1	1
4-5	A	1	1	1	1	1	1	1
	B	1	1	1	1	1	1	1

5-6	A	1	1	1	1	1	1	1
	B	1	1	1	1	1	1	1
6-7	A	0	0	0	0	0	1	1
	B	1	1	1	1	1	1	1
7-8	A	1	1	1	1	1	0	0
	B	1	1	1	1	1	1	1
8-9	A	1	0	1	0	1	1	1
	B	1	1	1	1	1	1	1
9-10	A	0	0	0	0	1	0	0
	B	1	1	1	1	0	1	1
10-11	A	0	0	0	0	1	0	0
	B	1	1	1	1	0	1	1
11-12	A	0	0	0	1	0	0	0
	B	0	0	0	0	0	0	0
12-13	A	0	1	0	1	0	0	1
	B	0	0	0	1	0	0	0
13-14	A	0	0	0	1	0	1	1
	B	0	0	0	1	0	1	0
14-15	A	0	0	1	0	0	0	0
	B	1	0	1	1	1	1	1
15-16	A	1	0	1	0	0	1	0
	B	0	1	0	0	1	1	1
16-17	A	1	0	1	0	1	1	1
	B	0	1	0	0	1	1	1
17-18	A	1	1	1	0	1	1	1

	B	0	1	0	1	0	0	1
18-19	A	0	1	0	1	1	0	1
	B	1	0	1	0	0	0	0
19-20	A	0	0	1	1	0	0	0
	B	1	0	0	1	0	1	0
20-21	A	1	0	1	1	1	1	1
	B	1	0	0	1	1	1	0
21-22	A	1	1	1	1	1	1	1
	B	1	1	1	1	1	1	0
22-23	A	1	1	1	1	1	1	1
	B	1	1	1	1	1	1	1
23-0	A	1	1	1	1	1	1	1
	B	1	1	1	1	1	1	1

Week 4:

Time	Users\Day	Mon	Tue	Wed	Thu	Fri	Sat	Sun
0-1	A	1	1	1	1	1	1	1
	B	1	1	1	1	1	1	1
1-2	A	1	1	1	1	1	1	1
	B	1	1	1	1	1	1	1
2-3	A	1	1	1	1	1	1	1
	B	1	1	1	1	1	1	1
3-4	A	1	1	1	1	1	1	1
	B	1	1	1	1	1	1	1

4-5	A	1	1	1	1	1	1	1
	B	1	1	1	1	1	1	1
5-6	A	1	1	1	1	1	1	1
	B	1	1	1	1	1	1	1
6-7	A	0	0	0	0	0	1	1
	B	1	1	1	1	1	1	1
7-8	A	1	1	1	1	1	0	0
	B	1	1	1	1	1	1	1
8-9	A	0	1	1	1	1	1	1
	B	1	1	1	1	1	1	1
9-10	A	0	0	1	0	0	0	1
	B	1	1	1	1	1	1	1
10-11	A	0	0	0	0	0	0	0
	B	0	1	0	1	0	1	1
11-12	A	0	0	0	0	0	0	0
	B	0	0	0	0	0	1	1
12-13	A	0	0	0	0	0	0	0
	B	0	0	0	1	0	0	0
13-14	A	1	1	0	0	0	0	1
	B	0	0	1	1	0	0	1
14-15	A	1	1	0	0	1	1	0
	B	0	1	1	0	0	0	1
15-16	A	0	0	1	0	1	1	1
	B	1	1	1	0	0	0	1
16-17	A	0	0	1	1	1	1	0

	B	1	1	1	0	1	0	0
17-18	A	1	0	1	1	0	1	1
	B	1	0	0	1	0	1	0
18-19	A	1	1	1	0	0	0	0
	B	0	1	0	1	0	0	0
19-20	A	0	1	0	0	0	0	1
	B	1	1	0	0	1	1	1
20-21	A	1	1	0	1	0	1	1
	B	1	1	0	0	1	1	1
21-22	A	1	1	1	1	0	1	1
	B	1	1	1	1	1	1	1
22-23	A	1	1	1	1	0	1	1
	B	1	1	1	1	1	1	1
23-0	A	1	1	1	1	1	1	1
	B	1	1	1	1	1	1	1

New database format:

day	**hour**	**user_id**	**count**	**week**
Mon	8-9	A	1	1
Mon	8-9	A	0	2
Mon	8-9	A	1	3
Mon	8-9	A	0	4