Experiment - 5

Aim

Create secure, production-ready **RESTful APIs**

What We Have Done in This Project

1. **Backend Setup**

   ○ Node.js with **Express** as the backend framework.

   ○ **MongoDB (Atlas)** used as the database, connected securely via a .env file.

   ○ **Mongoose** ODM for database modeling.

2. **Security Features**

   ○ **JWT Authentication**: Every request is verified using JSON Web Tokens.

   ○ **Role-Based Authorization**: Certain routes (e.g., task creation, update, deletion) are accessible only to authorized users.

   ○ **Environment Variables**: Sensitive information like MONGO_URI and JWT_SECRET are stored in .env.

3. **RESTful API Design**

   ○ Standard CRUD routes implemented for tasks:

     ■ **GET** /api/task/all – Fetch all tasks (protected route)

     ■ **GET** /api/task/:id – Fetch a single task

     ■ **POST** /api/task/add – Create a new task (protected route)

     ■ **PUT** /api/task/:id – Update task details (protected route)

     ■ **DELETE** /api/task/:id – Delete a task (protected route)

- - **PUT** /api/task/category/:id – Update task category (protected route)

  - **Controllers** handle the logic while **routes** define the endpoints.

  - **Middlewares**:

    - authorization.js – Checks user role and permission

    - wrapAsync.js – Handles async errors globally

4. **Validation & Data Integrity**

   - Task schema (models/Task.js) enforces required fields: title, priority, category, userName.

   - Ensures correct data types and unique constraints.

5. **Frontend Integration**

   - React frontend consumes these RESTful APIs.

   - Tasks can be added, updated, deleted, or categorized.

   - JWT token is stored in frontend to authenticate API requests.

   - Real-time updates and notifications via React components.

6. **Production-Ready Features**

   - Server ready to run on **PORT 3000** or any environment variable.

   - Environment-based configuration using .env.

   - Secure database connection to MongoDB Atlas.

   - Proper error handling for invalid routes and server errors.

Backend :

A] Code :

a) Server Setup (server.js)

- Show how **Express is configured**, **CORS setup**, **middleware**, and **database connection**.

- Include .env usage with process.env.MONGO_URI to highlight security.

```
task_management > backend > JS server.js > [∅] corsOptions
1    const dotenv = require("dotenv");
2    dotenv.config();
3    const express = require("express");
4    const cors = require("cors");
5    const mongoose = require("mongoose");
6
7    const app = express();
```

```
app.use(cors(corsOptions));
app.use(express.json());
```

```
// Connect to Database
main()
    .then(() => console.log("Database Connection established"))
    .catch((err) => console.log(err));

async function main() {
    await mongoose.connect(process.env.MONGODB_URI);
}
```

b) Authentication Route (routes/auth.js)

- Include **login and registration endpoints**, showing JWT issuance.

```
task_management > backend > routes > JS auth.js > ...
  1    const express = require("express");
  2    const router = express.Router();
  3    const authControllers = require("../controllers/auth");
  4    const wrapAsync = require("../middlewares/wrapAsync");
  5
  6    router.post("/register", wrapAsync(authControllers.registerUser));
  7    router.post("/login", wrapAsync(authControllers.loginUser));
  8
  9    module.exports = router;
 10    |
```

c) Protected Task Routes (routes/task.js)

- Show **how authorization middleware is used** for secure endpoints.

```
router.get("/all", authorization, wrapAsync(getAllTask));
router.get("/:id", wrapAsync(getTask));
router.post("/add", authorization, wrapAsync(addTask));
router.put("/:id", authorization, wrapAsync(updateTask));
router.delete("/:id", authorization, wrapAsync(deleteTask));
router.put("/category/:id", authorization, wrapAsync(updateCategory));
```

d) Task Model (models/task.js)

- Show **Mongoose schema**, especially required fields like title, priority, category, userName.

```javascript
const mongoose = require("mongoose");

const taskSchema = new mongoose.Schema(
    {
        title: {
            type: String,
            required: true,
            unique: true,
            trim: true,
        },
        priority: {
            type: String,
            required: true,
        },
        category: {
            type: String,
            required: true,
            default: "to-do",
        },
        checklist: [
            {
                name: {
                    type: String,
                    trim: true,
                    required: true,
                },
                isDone: {
                    type: Boolean,
                    default: false,
                    required: true,
                },
            },
        ],
```

```
34          userName: {
35              type: mongoose.Schema.ObjectId,
36              ref: "User",
37              required: true,
38          },
39          assign: String,
40
41          dueDate: Date,
42      },
43      {
44          timestamps: true,
45      }
46  );
47
48  const Task = mongoose.model("Task", taskSchema);
49  module.exports = Task;
50
```

```
Run npm audit for details.
PS D:\SEM5\sem5_FSD\fsd_5thexp\task_management\backend> npm run dev

> backend@1.0.0 dev
> nodemon server.js

[nodemon] 3.1.10
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,cjs,json
[nodemon] starting `node server.js`
Server listening on 3000
Database Connection established
```

<u>**Postman API request + response snippet**</u>

Why it's necessary

1. **Shows that your backend works:**
   A proof that your RESTful APIs are functional. Screenshots from Postman act as evidence.

2. **Captures API input/output:**
   You can show the **request payload** (JSON sent to API) and the **response** (JSON returned). This is critical for documentation.

3. **Validates security features:**
   Testing with JWT authentication shows that **only authorized users** can perform actions like creating or deleting tasks.

4. **Required for report completeness:**
   Most experiments require **code + output images**. Without testing APIs, your report will be incomplete.

http://localhost:3000/api/auth/login Save Share </>
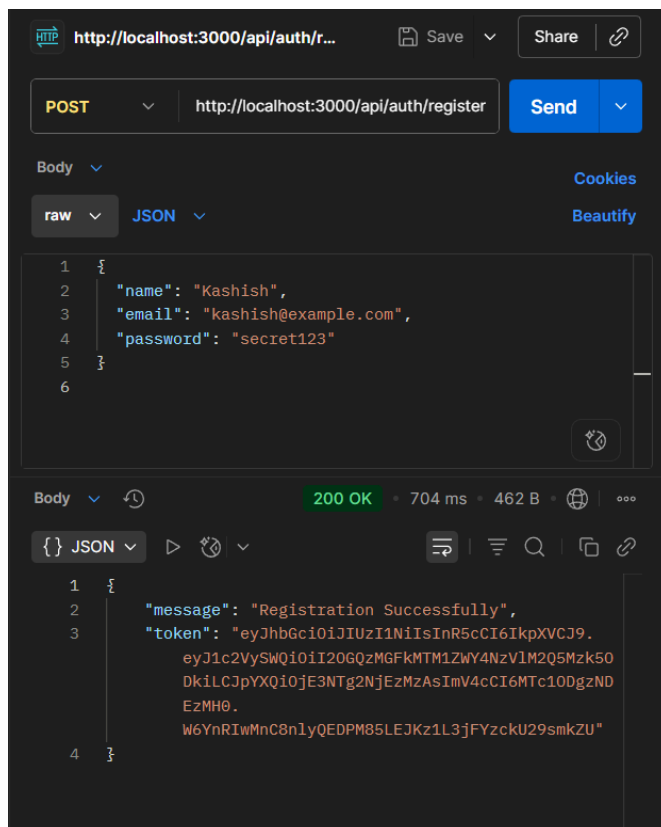
POST http://localhost:3000/api/auth/login **Send**

Body

raw JSON Cookies Beautify

```json
1  {
2    "email": "kashish@example.com",
3    "password": "secret123"
4  }
```

Body **200 OK** • 118 ms • 658 B

{ } JSON

```json
1  {
2      "message": "Login Successfully",
3      "data": {
4          "_id": "68d30ad135ef875e3d939989",
5          "name": "Kashish",
6          "password": null,
7          "email": "kashish@example.com",
8          "board": [],
9          "createdAt": "2025-09-23T21:02:10.004Z",
10         "updatedAt": "2025-09-23T21:02:10.004Z",
11         "__v": 0
12     },
13     "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
           eyJ1c2VySWQiOiI2OGQzMGFkMTM1ZWY4NzVlM2Q5Mzk5O
           DkiLCJpYXQiOjE3NTg2NjE0MDcsImV4cCI6MTc1ODgzND
           IwN30.
           Bf7g207b9hQIiMnai2GvbKeiDPTEQNXjmuNHdRNLdVs"
```

http://localhost:3000/api/task/all Save Share </>

GET http://localhost:3000/api/task/all **Send**

Params Authorization Headers (7) Body Scripts Settings Cookies

Headers 6 hidden

	Key	Value	Description	Bulk Edit	Presets
☑	Authorization	Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2V...			
	Key	Value	Description		

Body ⌄ 🕓 200 OK • 248 ms

{} JSON ⌄   ▷ Preview   ✳️ Visualize ⌄   ⇥

```
 1  {
 2      "message": "success",
 3      "data": {
 4          "backlog": [],
 5          "todo": [
 6              {
 7                  "_id": "68d30e2635ef875e3d93998f",
 8                  "title": "Finish FSD Experiment",
 9                  "priority": "high",
10                  "category": "to-do",
11                  "checklist": [],
12                  "userName": {
13                      "_id": "68d30ad135ef875e3d939989",
14                      "name": "Kashish"
15                  },
16                  "createdAt": "2025-09-23T21:16:22.577Z",
17                  "updatedAt": "2025-09-23T21:16:22.577Z",
18                  "__v": 0
19              }
20          ],
21          "inProgress": [],
22          "done": []
23      }
24  }
```

🌐 http://localhost:3000/api/task/68d30ad135ef875e3d939989   💾 Save ⌄   Share 🔗   </>

GET ⌄   http://localhost:3000/api/task/68d30ad135ef875e3d939989   Send ⌄

Params Auth Headers (7) Body Scripts Settings   Cookies

Headers 👁 6 hidden

	Key	Value	D...	⠇ Bulk Edit Presets ⌄
☑	Authorization	Bearer eyJhbGciOiJIUzI1NiIsIn...		
	Key	Value		Description

Body ⌄ 🕓   200 OK • 88 ms • 268 B 🌐 ⠇

{} JSON ⌄   ▷ Preview   ✳️ Visualize ⌄   ⇥ ≡ 🔍 ⎘ 🔗

```
1  {
2      "message": "success",
3      "data": null
4  }
```

http://localhost:3000/api/task/68d30e2635ef875e3d93998f

Save

Share

PUT   http://localhost:3000/api/task/68d30e2635ef875e3d93998f   Send

Params   Auth   Headers (9)   Body •   Scripts   Settings

raw   JSON

```json
1  {
2      "title": "Finish FSD Experiment Updated",
3      "priority": "medium"
4  }
5
```

Body   200 OK • 168 ms • 543 B

{} JSON   ▷ Preview   Visualize

```json
1  {
2      "message": "success",
3      "data": {
4          "_id": "68d30e2635ef875e3d93998f",
5          "title": "Finish FSD Experiment Updated",
6          "priority": "medium",
7          "category": "to-do",
8          "checklist": [],
9          "userName": {
10             "_id": "68d30ad135ef875e3d939989",
11             "name": "Kashish"
12         },
13         "createdAt": "2025-09-23T21:16:22.577Z",
14         "updatedAt": "2025-09-23T21:23:48.664Z",
15         "__v": 0
16     }
17 }
```

---

http://localhost:3000/api/task/68d30e2635ef875e3d93998f

Save

Share

DELETE   http://localhost:3000/api/task/68d30e2635ef875e3d93998f   Send

Params   Auth   Headers (7)   Body   Scripts   Settings   Cookies

Headers   👁 6 hidden

	Key	Value	D...	••• Bulk Edit Presets ∨
☑	Authorization	Bearer eyJhbGciOiJIUzI1NiIsIn...		
	Key	Value	Description	

Body   200 OK • 30 ms • 268 B

{} JSON   ▷ Preview   Visualize

```json
1  {
2      "message": "success",
3      "data": null
4  }
```

http://localhost:3000/api/task/add

POST   http://localhost:3000/api/task/add   Send

Body
raw   JSON   Beautify

2   "title": "Finish FSD Experiment",
3   "priority": "high",
4   "category": "to-do",
5   "userName": "68d30ad135ef875e3d939989"
6 }
7

Body   200 OK • 315 ms • 533 B

{} JSON

1 {
2   "message": "success",
3   "data": {
4   "_id": "68d30e2635ef875e3d93998f",
5   "title": "Finish FSD Experiment",
6   "priority": "high",
7   "category": "to-do",
8   "checklist": [],
9   "userName": {
10   "_id": "68d30ad135ef875e3d939989",
11   "name": "Kashish"
12   },
13   "createdAt": "2025-09-23T21:16:22.577Z",
14   "updatedAt": "2025-09-23T21:16:22.577Z",
15   "__v": 0

KASHISH'S ORG - 2025-04-02 > FSD_5TH > DATABASES

ClusterO   VERSION 8.0.13   REGION AWS Mumbai (ap-south-1)

Overview   Real Time   Metrics   Collections   Atlas Search   Query Insights   Performance Advisor   Online Archive   Cmd Line Tools   Infrastructure As Code

DATABASES: 1   COLLECTIONS: 2   PREVIEW New Data Explorer   VISUALIZE YOUR DATA   REFRESH

+ Create Database

Search Namespaces

FSD_5th
    tasks
    users

FSD_5th.users

STORAGE SIZE: 20KB   LOGICAL DATA SIZE: 205B   TOTAL DOCUMENTS: 1   INDEXES TOTAL SIZE: 40KB

Find   Indexes   Schema Anti-Patterns   Aggregation   Search Indexes

Generate queries from natural language in Compass   INSERT DOCUMENT

Filter   Type a query: { field: 'value' }   Reset Apply Options ▶

QUERY RESULTS: 1-1 OF 1

    _id: ObjectId('68d30ad135ef875e3d939989')
    name : "Kashish"
    password : "$2a$08$sqcGKeIDLRQJlt.envMms7uVk.kGFyQqZYXlATBemi2djXPQOYqLOy"
    email : "kashish@example.com"
    board : Array (empty)
    createdAt : 2025-09-23T21:02:10.004+00:00
    updatedAt : 2025-09-23T21:02:10.004+00:00
    __v : 0

30% extra :

Task enhancements: categories, priority levels, checklists, and assignments

Analytics endpoints for task statistics

Frontend integration: React + Vite with JWT token handling

Real-time updates reflected on the UI

React-Redux for state management

Environment-based config using .env

Frontend

```
∨  frontend
  >   node_modules
  ∨   src
    >   assets
    >   components
    >   css
    >   hooks
    >   pages
    >   redux
    >   utils
        App.jsx
        index.css
        main.jsx
      .eslintrc.cjs
      .gitignore
      index.html
      package-lock.json
      package.json
```
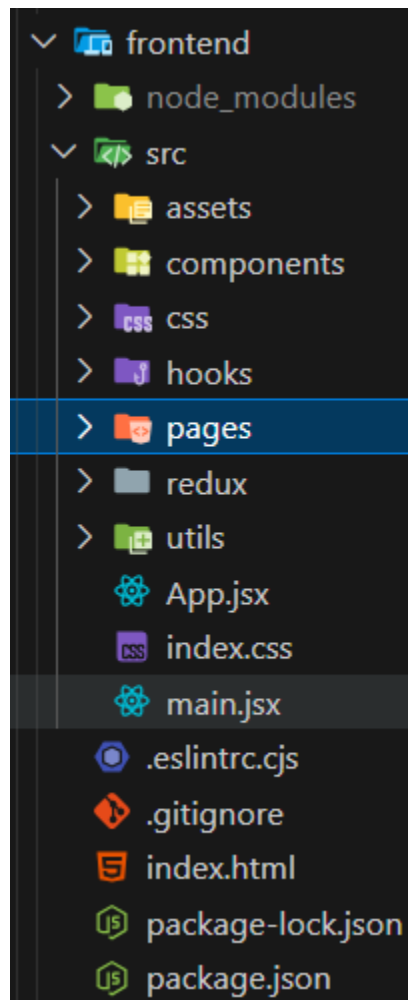
```
PS D:\SEM5\sem5_FSD\fsd_5thexp\task_management\frontend> npm install

npm warn deprecated inflight@1.0.6: This module is not supported, an
d leaks memory. Do not use it. Check out lru-cache if you want a goo
d and tested way to coalesce async requests by a key value, which is
 much more comprehensive and powerful.
npm warn deprecated rimraf@3.0.2: Rimraf versions prior to v4 are no
 longer supported
```

```
PS D:\SEM5\sem5_FSD\fsd_5thexp\task_management\frontend> npm run dev

> frontend@0.0.0 dev
> vite


  VITE v5.2.12  ready in 252 ms

  →  Local:   http://localhost:5173/
  →  Network: use --host to expose
  →  press h + enter to show help
```

Needs 1 Special Char ✕

Register

👤 Kashish Chhabadiya

✉ 2023.kashish.chhabac

🔒 ••••••••••• 👁

🔒 ••••••••••• 👁

Register

Have an account ?
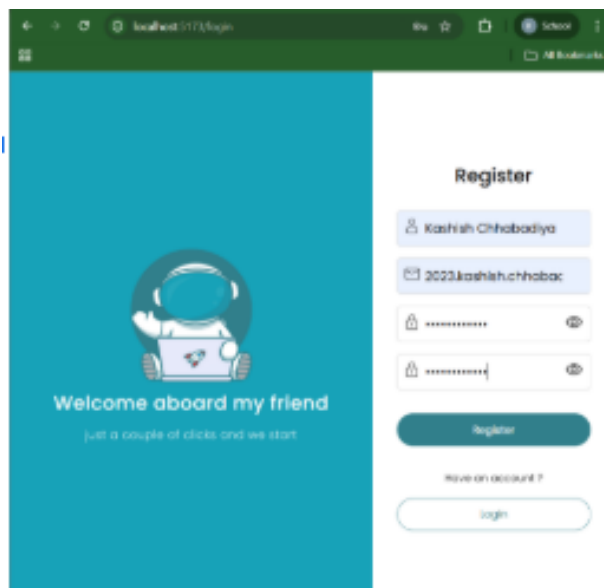
Login

Welcome aboard my friend

just a couple of clicks and we start

Login

✉ 2023.kashish.chhabac

🔒 ••••••••• 👁

Login

Have no account yet?

Register

Settings

Pro Manage

- Board
- Analytics
- Settings

Kashish Chhabadiya

2023.kashish.chhabadiya@ves.ac.in

Old Password 👁

New Password 👁

Update

Creation of tasks :

Title *

Room cleaning

Select Priority * 🔴 HIGH PRIORITY 🔵 MODERATE PRIORITY 🟢 LOW PRIORITY

Checklist (0/0) *

+ Add New

20/20/2025   Cancel   Save

Welcome! Kashish

Board ⚆ Add People

24th Sep, 2025

This month ⌄

Board

Analytics

Settings

Log out

Backlog

To Do

● LOW PRIORITY KC ...
Assignment fsd
Checklist (0/1) ⌄
BACKLOG PROGRESS DONE

● HIGH PRIORITY KC ...
Presentation dmbi
Checklist (1/2) ⌄
BACKLOG PROGRESS DONE

● MODERATE PRIORITY KC ...
Room cleaning
Checklist (2/2) ⌄
Sep 20 BACKLOG PROGRESS DONE

● LOW PRIORITY KC ...
Skincare
Checklist (0/1) ⌃

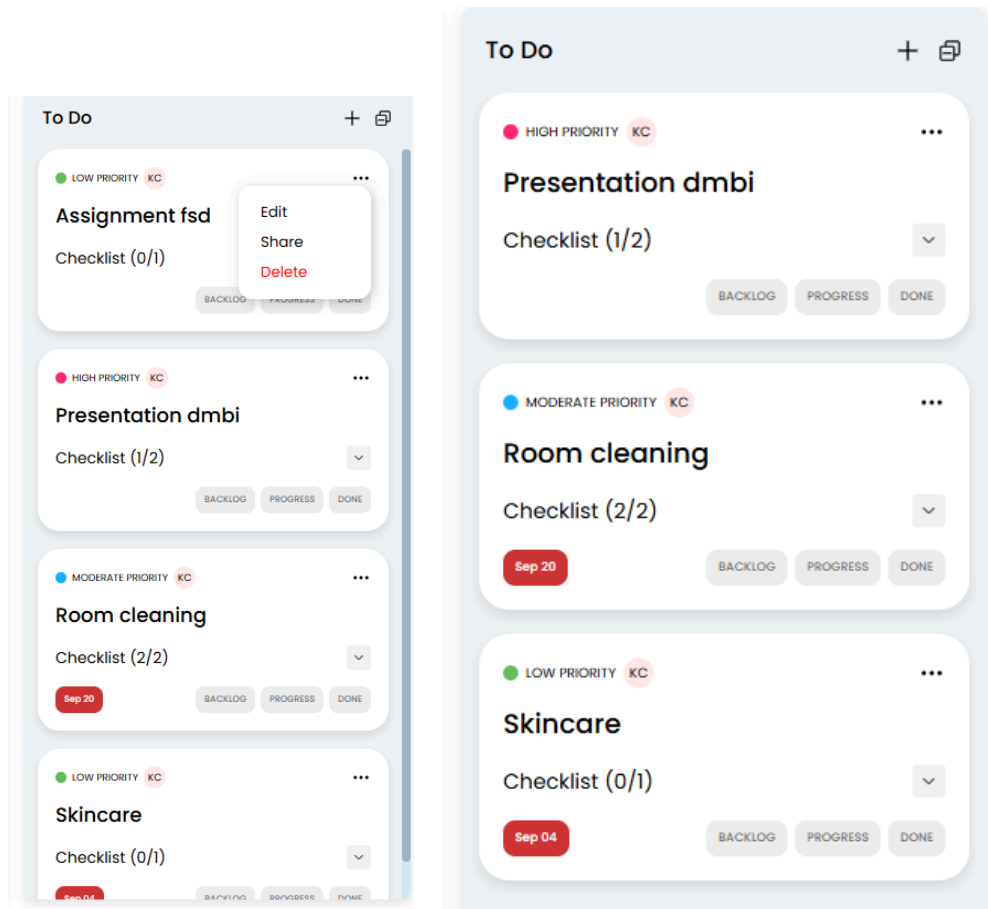**In Progress**

Done

●
Pr
Cl

Add people to the board

Enter the email

2023.kashish.chhabadiya@ves....

2024.gunjan.athwani@ves.ac.in

Add Email

● MODERATE PRIORITY KC

Room cleaning

Delete :

Editing tasks :
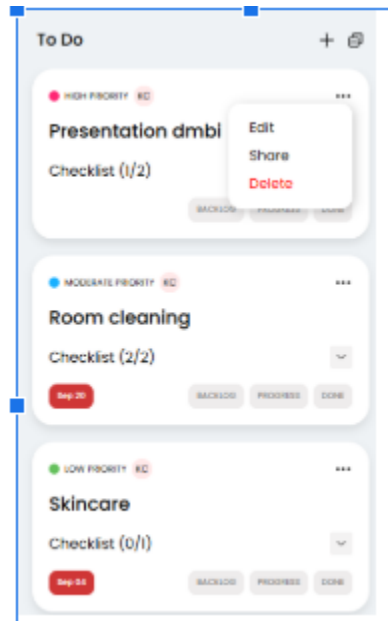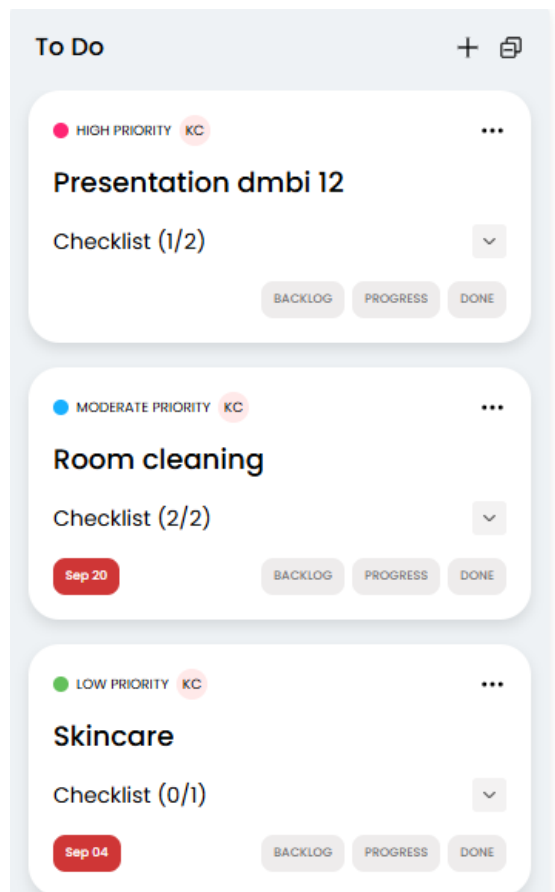
Update task status :
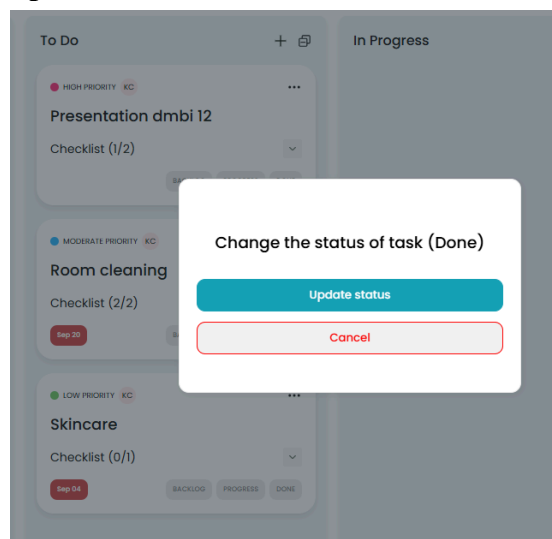
Welcome! Kashish

Board   Add People

This week ⌄

Backlog	To Do	In Progress	Done

To Do + ⧉

● MODERATE PRIORITY KC   ⋯
Room cleaning
Checklist (2/2) ⌄
Sep 20   BACKLOG   PROGRESS   DONE

● LOW PRIORITY KC   ⋯
Skincare
Checklist (0/1) ⌄
Sep 04   BACKLOG   PROGRESS   DONE

Done ⧉

● HIGH PRIORITY KC   ⋯
Presentation dmbi 12
Checklist (1/2) ⌄
BACKLOG   TO-DO   PROGRESS