

Project Based Assignment CA-3 (INT-219)

Task Manager API

A project Report

Submitted in partial fulfillment of the requirements for the award of
degree of

Bachelor of Technology

Computer Science and Engineering

Submitted To

**LOVELY PROFESSIONAL UNIVERSITY
PHAGWARA , PUNJAB**



L OVELY
P ROFESSIONAL
U NIVERSITY

Name of Student : Sudeep Kumar

Registration Number: 11917205

Section : KM017

Roll Number: RKM017A39

Contents

S. No	Content	Page No
01.	Introduction	3-3
02.	Technology Used	4-6
03.	Description	7-10
04.	Screenshots	11-12
05.	Code/Link	12-12
06.	Conclusion	13-13
07.	References	14-14

Introduction to the project

This project is a part of the cumulative Assessment under the course code of INT 219 (Front End Web Development) as provided by the Lovely Professional University in the academic curriculum. This project is of a Basic Task Manager API build with the help of HTML (Hyper Text Markup Language), CSS (Cascading Style Sheet) and JavaScript as for front end and Node JS, Express and Mongo DB as for the backend development.

This is a task manager API where the user can able to store or save the task he or she is planning to do as a planner and can able to view it for further reference and can able to work according to that. Also, once the user done with the task he or she can able to edit the task as completed and mark it as completed. If the user want to edit some details in the task he or she can also able to edit the task like renaming the task if necessary. User can also able to delete the task once completed from the list. This gives the user full liberty to keep track of the schedule he/she made earlier. The whole process has handled by the backend developed in Node JS and Express using some endpoints.




This application of this project is very handy in the real life , as in the real life it is very important to schedule our works according to time , in this way we can able to utilize the time in proper and effective way, this application help us to do exactly all these works hence it help us to save our time for the other useful things.

The detailed description and implementation of whole project is provided in the further coming sections. So let's deep dive in the further sections for complete understanding of the project.

Technology Used in the Project

Here we discuss about the various technologies used for the successful completion of this project. We discuss what technologies we used for the backend and frontend development in details.

Following are the technologies are used in this project:

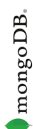
-  **HTML (Hyper Text Markup Language)** : It is a markup language used to design web pages and can be understood by HTTP or HTTPS Protocols. It consists of number of tags which is used to give the basic structure to the web pages. It is also very much demanding in the real world as in this modern world, we can say most of the things are based on internet and websites and it is a building block of any websites. Facebook, Twitter, Amazon are some of the big websites using this technology. In this project it is again used as building block of our webpages.
-  **CSS (Cascading Style Sheets)** : It is a language used in websites to style the webpages and give them attractive looks. It is used to give every element of style we can think of to the webpages like animation, transition , designing , etc.... All the design in this project is created by using this CSS . In real world again this is also very useful. You can go to any website and you see you will find it attractive , there are various transition effect and also various animations . You see different styled button, links , Nav-bars , etc. This all is done with the help of the CSS.
-  **JavaScript** : It is again a language or more correctly a scripting language . which is used to manipulate the webpages dynamically. It is used to add functionality to different element of the webpages like buttons , links etc. It is used to make our website work and function, each and every thing our website do is done by JavaScript and its used concept of DOM (Document Object Module) to do all sorts of things. If we don't use this our webpages is just like a static image and we will not be able to do any operation in it. Our whole project implement this JavaScript for various functionalities of buttons, links or to do any request form front end to backend . In real world again it is very useful to make things work in the web technologies.



- **Node JS** : It is one of the newly originated technologies which is in use from recent decades. It help us to integrate the JavaScript into the Backend Development. In previous time we used different technologies like php , .NET (Dot NET) etc. to create backend but Node JS help us to do it by using just the JavaScript and it become very useful because we donot need to learn various languages for it. Node JS is the bundle of we can say the library of the JavaScript oriented logics which kept on single place and are used to do backend development . In this project Node JS is used to make the backend and the server side logic. And now a days in real world its demand is increasing because it is very useful as it give ease to the developer. And also it provide a complete node module to do different tasks which seems to be very difficult if applied manually and all these things are managed by NPM(node package manager).



- **Express** : Express is just a module come with Node JS which help us to do request in a easy way and also in an efficient way. Without using express we can also do stuffs but that become little lengthy and complex as we have to take care of different things. Express give us ease and provide some inbuilt functions to do all the same tasks in a very simple manner. Hence it is very effective and developer friendly and it also saves our time. In Current world wherever the Node JS is being used express is the first choice of the developer as it give them that ease to do things and manipulate some of the complex things while developing.



- **Mongo DB** : It is a technology which deals with the database. It is a database management system which help use to store data in a particular format in their cloud storage and let us do whatever operation we want to do with the stored data. Some of the basic operations are Create, Read, Update, Delete called as CRUD . It is also called No SQL database since it donot implement the relational database logics means it donot based on table creation . It Is a database Based on JSON logic, means it stored all the information in the list of JSON Objects and let us use those data. It Provide us a cloud space called Mongo DB Atlas where we can store the data. In this project all the data are stored in Mongo DB database which can be seen using Mongo DB Atlas. In the real world it is a emerging Database system and now a days many companies is shifting towards this as per their requirements.

- **Mongoose** : Mongoose is a Object Data Modelling(ODM) library for Mongo DB and Node. It manages the relationships between the data , provides schemas validation, and is used to translate between objects in code and the representation of those object in mongo DB. Basically it is the library which help us to store the data in mongo DB but in a efficient way . It internally create all the data given to it to the format in which data is stored in Mongo DB so we donot need to worry about the format of storing data or in other words donot need to format the data in required format manually mongoose handle it smoothly. It provides various function to do manipulation with the database which make our life simple . In this project complete database model is based on this mongoose and the function offered by this is used to do operations . In real world again it is in demand as it give ease to do the things efficiently.

- **Bootstrap**: It is a framework which help us to design websites faster and easier . It includes HTML and CSS based design templates for typography, forms, buttons, tables, navigation, modals, image carousels, etc . It also give support for JavaScript plugin . It also let us do some external changes if we want to in the designing which is offered by it by default. In this project some of the bootstrap is there which help in the design of some of the components there. In real world since it give inbuilt design template to the user it is very much useful. Since developer donot need to do all things with scratch they just need to take the template and modify it as per their requirement and need.

- **Axios** : It is just a JavaScript module or library used to do request to the server like GET, POST, PUT, PATCH , DELETE and will give use the information from the server and will also provide information to the server. In this project all the request are done using this very library and it works in a very efficient manner and easy to use.

- **CDN (Content Delivery Network)** : CDN refers as Content Delivery Network is used to integrate the content from a particular web address to our webpages without actually downloading all the contents explicitly . With the help of CDN we can use the information present in that particular CDN address anywhere if we want and hence it keeps things simple. We donot need to import the complete file or information explicitly, it just provide all those information which we need nothing other than that. In our project we used CDN for axios and able to do request with the help of these CDN link. It is generally used in the real world organization because it keeps thing pretty simple and reduce the wastage of the computer storage.

Description of the Project

This section is all about the implementation part of the project, A detailed description of the front end as well as the backend is provided in this section individually. At first we discuss about the front end and then we discuss about the backend part.

Front End Description

In this part we discuss about the frontend where we implement all the frontend logic. Talking about the frontend at first, it contains a label named “Task Manager”. Now next there is a form which will take input of the task entered by the user and then after the submission of this task, there is a Task DOM which will render or update according to all the tasks provided by the user which is stored in the database from the backend.

In the Task DOM we can able to see all the tasks and with all the tasks 2 types of buttons are associated, the first is Edit Task button and the second is Delete Task button. The whole Task DOM is scrollable once it get overflowed. And also having styles.

Now if some user click edit Details button provided with all the tasks he /she will be redirected to the new page where there is a id of the task provided and there user can able to edit the task details , some of the details which can be editable by the user are follows:

- **Task Name :-** Here user can able to edit the task name if he /she intended to do .
- **Is completed / Not completed :** Here user can able to mark the task as completed or not as per his/her progress

There is a Button in the edit details window using which user can able to navigate to the main window and can able to see the updated details in the Task DOM.

If the user changes the name in the edit details the new name will be displayed there and if the user changes the status (completed) then he/she can able to see the completed task with a tick mark and stroked text in the Task DOM which means the task is completed.

Now if the user click the Delete button the task will be completely get deleted from there as well as the database and again the updated DOM will be displayed in the screen to the user.

All the styles and effect is done in the frontend with the help of CSS , like there is a background, box-shadow, color , border etc.. which is all applied there with the help of CSS Selectors

Talking about the dynamic nature all the front end interaction is handled with the help of JavaScript ,all the button click are handled with the help of **addEventListener()** which take an event and return a callback with respect to that event. In this project the mostly used event is click, for fetching the details from the API we use “axios” CDN , which is providing all the responses from the backend to the frontend so that we can able to manipulate it.

This is all about the frontend part application of the project now in the next part we discuss the backend part of the project.

Backend Description

In this part we will discuss about the backend implementation of the project there are some major parts of the backend part are as follows

- **The database Structure**
- **The end points**
- **The action on the endpoints**

Database Structure : Here we discuss about the database structure , in what format the data is stored in the database. Before this it is important to know that all the database part has handled with the help of Mongoose, Mongo DB so we will have all the schemas as per mongoose schemas.

Here we are storing two thing in the database the first thing is the Name of the task and the other thing is the status (completed) which means whether the task is completed or not, initially all the task is in false state. The snap of the schema is shown next.


```
const mongoose = require("mongoose");
const TaskmanagerSchema = new mongoose.Schema({
  name: {
    type: String,
    required: [true, "must provide a name"],
    trim: true,
    maxlength: [20, "name cannot be more than 20 characters"],
  },
  completed: {
    type: Boolean,
    default: false,
  },
});

module.exports = mongoose.model("Task", TaskmanagerSchema);
```

The endpoints: Now after the schemas we will discuss about the end points or the routes which are necessary for doing the backend processes with the database. The endpoints are implemented here with the help of the express. The required endpoint is as follows:

- The parent endpoint for all the endpoint : `app.use('/api/v1/tasks', tasks);`
- `router.route('/').get(getAllTasks).post(createTask);`
- `router.route('/:id').get(getTask).patch(updateTask).delete(deleteTask);`

Here the first endpoint is the parent endpoint which should be mentioned pre- all the endpoints.

The second endpoint is for the getting task and creating the task as per the given implementation

The third endpoint is for the getting single task, updating single task, and deleting single task.

The action of the End points : Here in this section we discuss about the operation which are going on the endpoints , or we can call it REST API operations Like GET, PUT, PATCH , POST and DELETE.

The controllers take the endpoint and perform the specific operations and do update in the database. The snap of one of the operation is provided below for the reference and all the operations are done in the similar fashion. The whole operation is used with the help of CRUD related function provided by Mongoose client which is responsible to do all the respective changes in the databases.

```
const getAllTasks = asyncWrapper(async (req, res) => {  
  const tasks = await Task.find({});  
  res.status(200).json({ tasks });  
});
```

Fig 3.2 : A sample controller for getting all task managed by async wrapper

Some of the function which are used for CRUD operations are follows :

- `Model.deleteMany()`
- `Model.deleteOne()`
- `Model.find()`
- `Model.findById()`
- `Model.findByIdAndDelete()`
- `Model.findByIdAndRemove()`
- `Model.findByIdAndUpdate()`
- `Model.findOne()`
- `Model.findOneAndDelete()`
- `Model.findOneAndRemove()`
- `Model.findOneAndReplace()`
- `Model.findOneAndUpdate()`
- `Model.replaceOne()`
- `Model.updateMany()`
- `Model.updateOne()`

Fig 3.3 : CRUD functions provided by Mongoose

This is the complete implementation part now in next section consist the screenshots of the interface for proper understanding.

Screenshots of the Interface

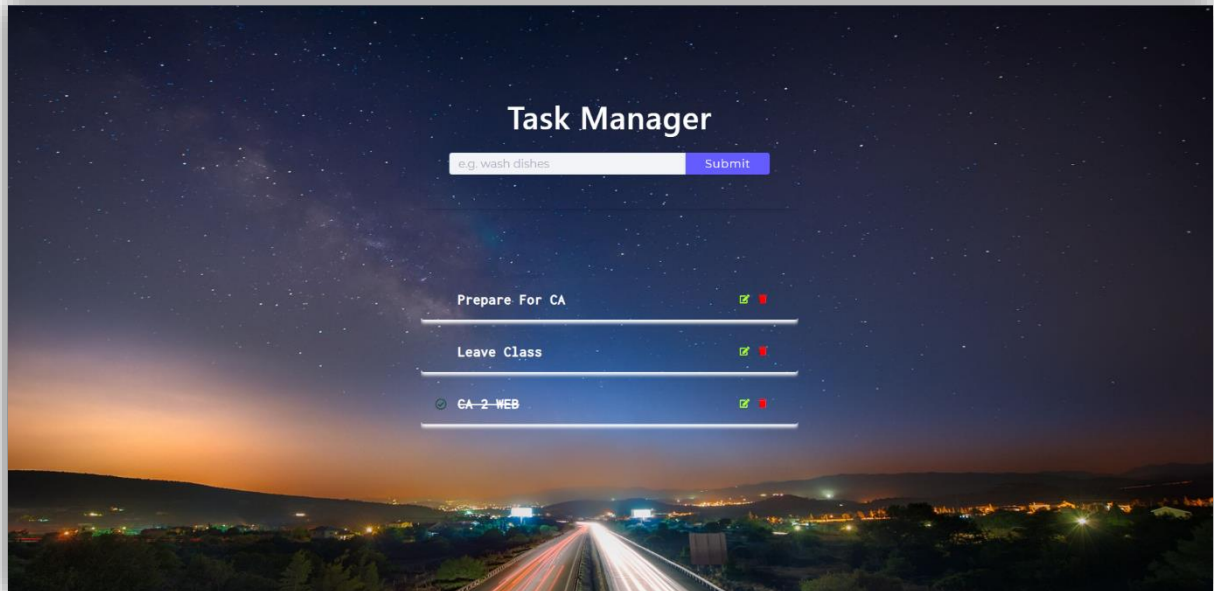


Fig 4.1 Front Page

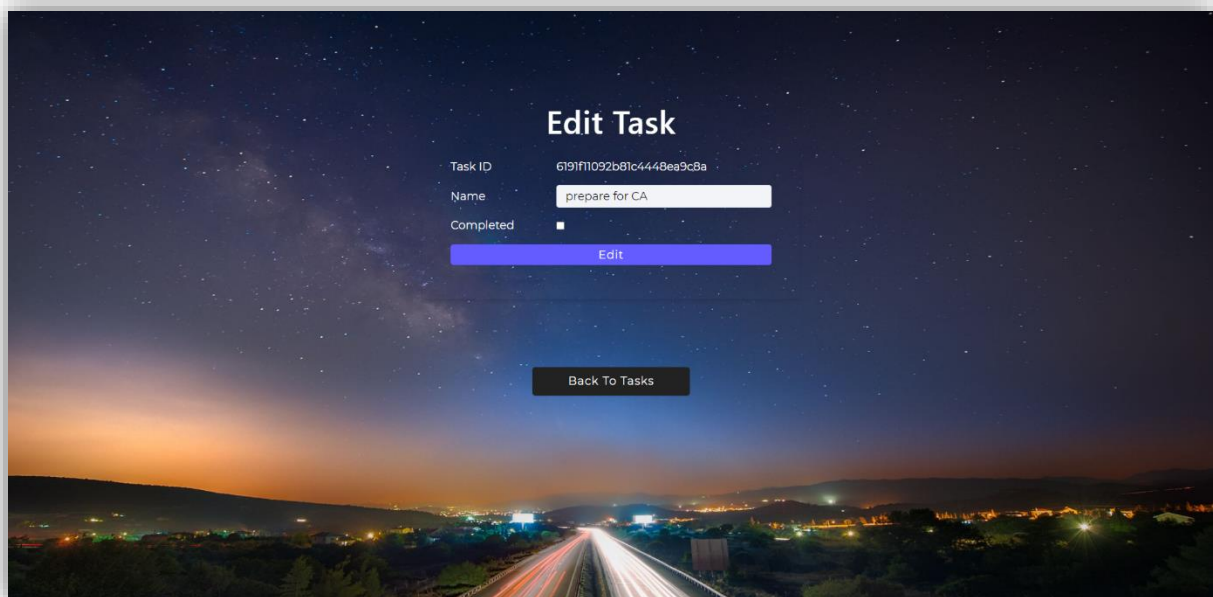


Fig 4.2 Edit Page



Fig 4.3 Data Representation In The database

The above are the screenshots of the all necessary interfaces which is very helpful for the understanding of the project. In this way we are done with the implementation part and in next section we conclude our project

CODE Github Repo Link:

<https://github.com/Sudeep-kr-1999/Task-Manager-.git>

Conclusion

This section is about concluding the project. This is all about the brief summary of the whole project, Some of the conclusions about the project which can be drawn is mentioned as follows:

- This project is based on the task manager API which is used to store the schedule or planning of work for the user.
- This project is developed with the help of HTML, CSS and JavaScript for the frontend interaction and behavior.
- For backend this project uses technologies like Node JS, Mongo DB , express for doing all the backend logics.
- This project help us to understand the way of designing the database schemas to store information in the efficient way.
- This project help us to learn about the REST API which stands for Representational State Transfer Application Programming Interface which is very important for the request and response operations in the backend.
- This project taught us about basic request like PUT, PATCH, GET, POST, DELETE which is very useful for manipulation data in the database and for the communication of the frontend with the backend.
- This project taught us about some of the important function used in the JavaScript for interaction with the user and manipulation of DOM
- This project also taught us about the proper styling of the webpage using CSS and like Transition, flex box, position, animation, transform etc..
- We can also conclude that this project is a great real life application as it is very useful for the proper utility of time.
- At last we can say that there is also a great scope of adding some more and useful features in this project and make it scalable.

References

This is the last part of this report and this is all about references. Below mentioned some of the references that are taken to make this project successful and form this project a thankful gesture to all the helper references which help us make this project successful.

- <https://NodeJS.org/en/docs/>
- <https://expressjs.com/>
- <https://expressjs.com/en/starter/hello-world.html>
- <https://docs.mongodb.com/>
- <https://docs.atlas.mongodb.com/getting-started/>
- <https://mongoosejs.com/>
- <https://mongoosejs.com/docs/guide.html>
- <https://mongoosejs.com/docs/queries.html>
- <https://mongoosejs.com/docs/connections.html>
- <https://www.jotform.com/form-templates/category/application-form>
- <https://developer.mozilla.org/en-US/docs/Web/CSS>
- <https://www.w3schools.com/js/DEFAULT.asp>
- <https://JavaScript.info/>
- <https://www.freecodecamp.org/>