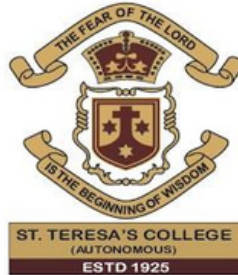


**ST. TERESA'S COLLEGE (AUTONOMOUS),
ERNAKULAM**

AFFILIATED TO MAHATMA GANDHI UNIVERSITY



**PROJECT REPORT ON
To-Do List**

In partial fulfillment of the requirements for the award of the degree of

**BACHELOR OF VOCATION IN
SOFTWARE DEVELOPMENT**

By

Trisha Tejniya M R

III B.Voc Software Development

Register No: VB21SWD026

Under the guidance of

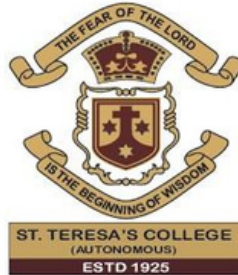
Ms. RAJI S PILLAI

Department of Computer Applications

2021-2023

**ST. TERESA'S COLLEGE (AUTONOMOUS),
ERNAKULAM**

AFFILIATED TO MAHATMA GANDHI UNIVERSITY



**PROJECT REPORT ON
To-Do List**

In partial fulfillment of the requirements for the award of the degree of

**BACHELOR OF VOCATION IN
SOFTWARE DEVELOPMENT**

By

Trisha Tejniya M R

III B.Voc Software Development

Register No: VB21SWD026

Under the guidance of

Ms. RAJI S PILLAI

Department of Computer Applications

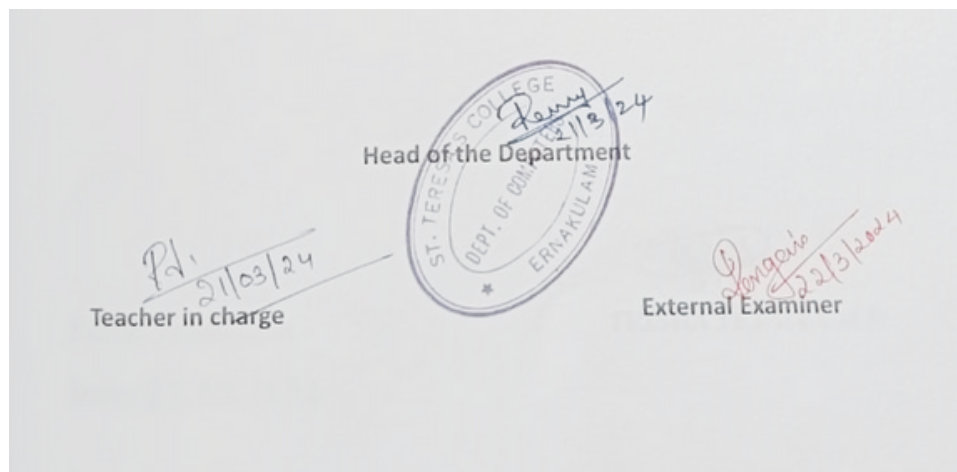
2021-2023



CERTIFICATE

This is to certify that the project report on **"To Do List Project"** is a bonafide record of the work done by **TRISHA TEJNIYA M R (VB21SWD026)** during the year 2021-2024 and submitted in partial fulfillment of the requirement for the degree of **B.Voc Software Development** of Mahatma Gandhi University

Submitted for end semester exam held on 22nd March 2024

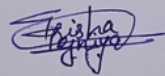


DECLARATION

I, **TRISHA TEJNIYA M R (Register no: VB21SWD026), B.Voc Software Development** final year student of St. Teresa's College (Autonomous), Ernakulam, hereby declare that the project submitted named **TO-DO LIST** for the Bachelors of Vocational Degree in Software Development is my original work. I further declare that the said work has not previously been submitted to any other university or academic body.

Place: Ernakulam

Date: 22.03.2024



TRISHA TEJNIYA M R

B.VOC SOFTWARE DEVELOPMENT

ACKNOWLEDGMENT

First and foremost, I thank God Almighty for his blessings. I take this opportunity to express my gratitude to all those who helped me in completing this project successfully. I wish to express my sincere gratitude to the

Provincial Superior and Manager Rev. Dr. Sr. Vinitha CSST,

Director Rev. Sr Emeline CSST, and Principal Dr. Alphonsa

Vijaya Joseph for providing all the facilities. I express my sincere

gratitude to the **Head of the department Ms. Remya C J** for the

support. I deeply express my sincere thanks to my guide **Asst. Prof.**

Ms. Raji S Pillai for her proper guidance and support throughout the

project work. I thank all other faculty of the department for their

exemplary guidance and appreciation throughout the project.

My deepest appreciation goes to all my friends and relatives for their support. I convey my hearty thanks to my parents and my brother who encouraged me and provided all the opportunities for the completion of this project

TRISHA TEJNIYA M R

Synopsis

This to-do list software offers a sleek and contemporary design, making task organization and scheduling effortless. Its active date and time feature allows users to set specific deadlines and reminders for each task, ensuring they stay on track and meet deadlines effectively. With intuitive navigation and a user-friendly interface, users can easily add, prioritize, and manage their tasks, enhancing productivity and organization in their daily lives or work projects.

CONTENTS

1. INTRODUCTION

2. SYSTEM ANALYSIS

2.1 Introduction.....	4
2.2 Existing System.....	4
2.3 Proposed System.....	4
2.4 Justification for the proposed system.....	4
2.5 Benefits of the proposed system.....	5
2.6 System specifications.....	5
2.7 Operating System.....	5
2.8 Language OR S/W package.....	5
2.9 H/W & S/W Specification(Minimum requirements).....	6

3. SYSTEM DESIGN

3.1 Introduction.....	8
3.2 Hierarchical Chart.....	8
3.3 E-R Diagram.....	9
3.4 Data Flow	

6. CONCLUSION

7. APPENDIX

8. BIBLIOGRAPHY

1. INTRODUCTION

To-do list software is a digital tool designed to help individuals and teams efficiently manage tasks, projects, and deadlines. These applications serve as an essential aid in organizing daily activities, prioritizing work, and tracking progress toward goals. Key features often include the ability to create and categorize tasks, set reminders, assign priorities, and sometimes collaborate with others by sharing lists or assigning tasks to team members. This software varies from simple, straightforward list-making apps to more complex systems that integrate project management functionalities, supporting both personal productivity and team-based workflows.

The appeal of to-do list software lies in its versatility and accessibility; many applications are available across various platforms, including web browsers, desktop applications, and mobile apps, ensuring users can keep track of their responsibilities anytime, anywhere. Customization options allow users to tailor the app's interface and functionality to suit their personal preferences and work styles, making task management more engaging and effective.

By consolidating tasks in one place, to-do list software helps reduce the cognitive load of remembering numerous deadlines and responsibilities, thus lowering stress levels and enhancing overall productivity. Whether for personal use, like managing daily chores and errands, or for professional settings, where teamwork and project deadlines are paramount, to-do list software has become an indispensable tool in the modern digital toolkit.

2. SYSTEM ANALYSIS

2.1 INTRODUCTION

System analysis for a to-do list software involves a detailed examination of various aspects related to its development, functionality, and user experience. The to-do list system aims to provide users with a digital tool for managing tasks and organizing their daily activities efficiently.

2.2 EXISTING SYSTEM

Todo List: Todo List is a popular task management app that allows users to create, organize, and prioritize tasks and projects.

2.3 PROPOSED SYSTEM

The website is designed for the students and we can have a clean design and intuitive user interface that prioritizes ease of use and efficiency. Implement a responsive design to ensure a seamless user experience across various devices and screen sizes.

2.4 JUSTIFICATION FOR THE PROPOSED SYSTEM

The justification for the outlined features on the webpage lies in their potential for a user-friendly interface, smart task management, and seamless integration with popular productivity tools. Its customization options allow users to tailor the application to their needs, while robust security measures ensure data protection and compliance. With advanced features like intelligent reminders and collaboration tools, the system enhances productivity and efficiency.

2.5 BENEFITS OF THE PROPOSED SYSTEM

It helps keep all tasks and responsibilities in one place, making it easier to stay organized and focused.

By prioritizing tasks and breaking down larger projects into smaller steps, users can tackle their work more efficiently.

Keeping track of all tasks in a centralized location can reduce the mental load and stress associated with trying to remember everything.

Many to-do list apps provide features to track the progress of tasks and projects, offering a sense of accomplishment as items are completed.

2.6 SYSTEM SPECIFICATIONS

System specification specifies the hardware and software configuration of the new system. It helps to define the operational and performance guidelines for system. Consumer Computing is a website developed in HTML, CSS, and Javascript for updating, viewing, and buying an electronic gadget.

2.7 OPERATING SYSTEM

Windows 11 is part of the series of operating systems developed by Microsoft. It is Microsoft's latest operating system, boasting a refreshed user interface, improved performance, and new features. Built-in Microsoft Teams integration enables seamless communication and collaboration. Gaming enhancements include DirectX 12 Ultimate support, Auto HDR, and DirectStorage for faster loading times. Accessibility improvements, like improved touch, pen, and voice input, ensure inclusivity. Security features such as Windows Hello and enhanced malware protection enhance system security. Windows 11 aims to elevate productivity, creativity, and connectivity for users across devices.

2.8 LANGUAGE OR S/W PACKAGE

HTML, CSS, and JavaScript are used to construct the webpage, utilizing Visual studio for coding purposes. Visual studio serves as a platform for writing code, while HTML, CSS, and JavaScript are employed respectively for structuring web content, styling elements, and adding interactivity and functionality to the webpage.

2.9 HARDWARE & SOFTWARE SPECIFICATION

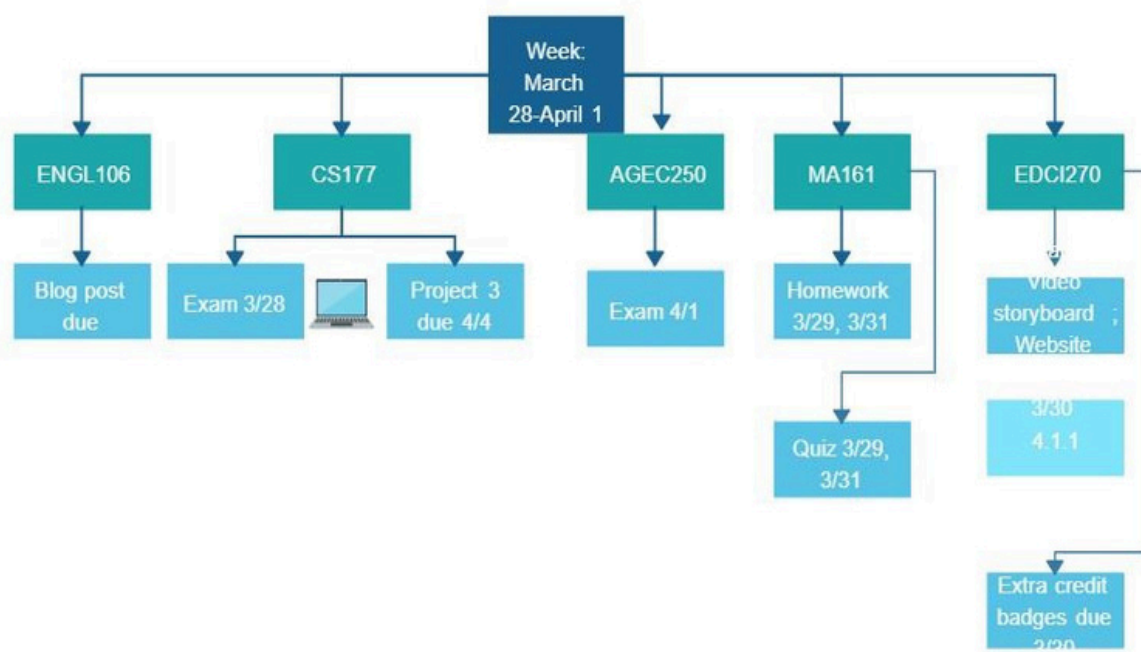
- > Front End: HTML, CSS, Javascript
- > Back End: Javascript
- > Operating System: Windows 11
- > Browser: Google Chrome

3. SYSTEM DESIGN

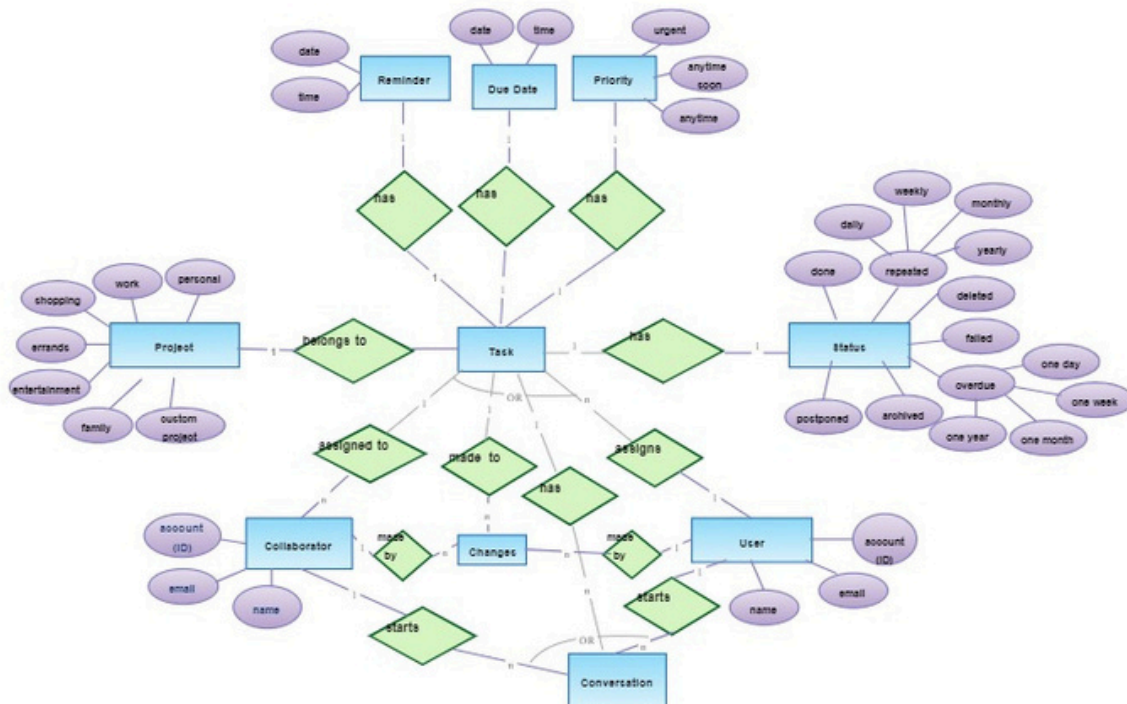
3.1 INTRODUCTION

The to-do list system design focuses on creating a user-friendly interface that enables efficient task management. Users can easily create, organize, and prioritize tasks, with options to set due dates and categorize tasks by project or priority. The system stores task data in a relational database, with attributes including title, description, due date, and status. Task organization is facilitated by algorithms that sort tasks based on priority and deadline, while smart reminders ensure timely completion. Integration with external services like calendars and email allows for seamless synchronization and collaboration. Security measures such as encryption and user authentication safeguard user data. The system is designed to be scalable, responsive, and adaptable to meet the evolving needs of users, providing a reliable and intuitive platform for task management.

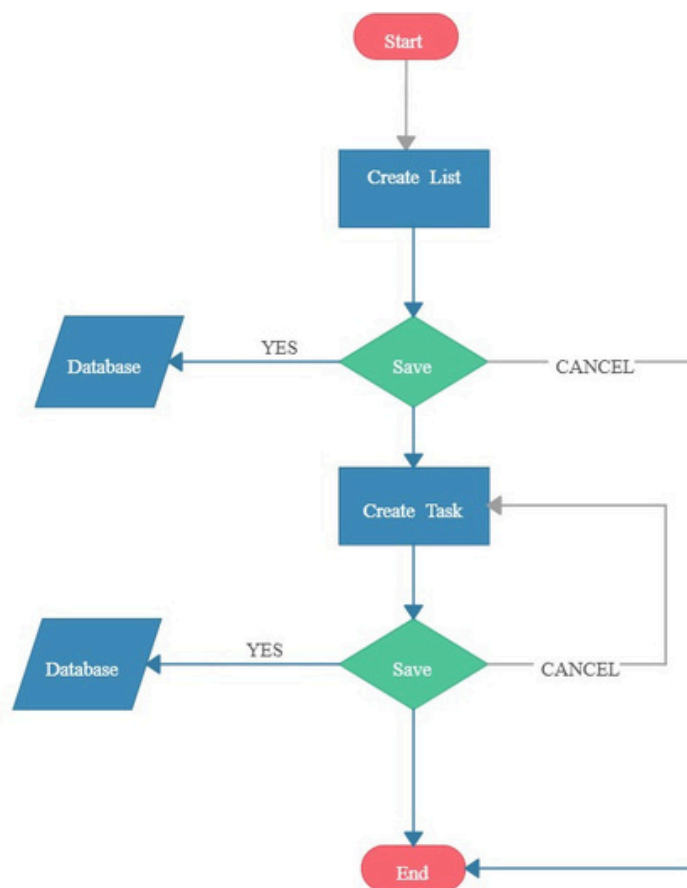
3.2 HIERARCHICAL CHART



3.3 E-R DIAGRAM



3.4 DATA FLOW DIAGRAM



3.5 DATA DICTIONARY

Task Table:

Field	Description
Task_ID	Unique identifier for the task
Title	Title or name of the task
Description	Additional details about the task
Due_Date	Deadline for completing the task
Priority	Priority level of the task
Status	Current status of the task

Category Table:

Field	Description
Category_ID	Unique identifier for the category
Name	Name or label of the category
Description	Additional details about the category

Reminder Table:

Field	Description
Reminder_ID	Unique identifier for the reminder
Task_ID	Reference to the associated task
Reminder_Date	Date and time for the reminder
Message	Reminder message or notification content
Status	Current status of the reminder

3.6 DATABASE DESIGN

Task Table:

Task_ID	Title	Description	Due_Date	Priority	Status
1	Study for Exam	Review chapters 3-5	2024-03-25	High	Incomplete
2	Grocery Shopping	Buy fruits and vegetables	2024-03-20	Medium	Incomplete
3	Call Mom	Wish her a happy birthday	2024-03-21	Low	Incomplete

Category Table:

Category_ID	Name	Description
1	Work	Tasks related to work
2	Personal	Personal tasks
3	Errands	Household errands

Reminder Table:

Reminder_ID	Task_ID	Reminder_Date	Message	Status
1	2	2024-03-20 09:00:00	Don't forget to buy groceries today!	Pending
2		2024-03-21 08:00:00	Remember to call Mom for her birthday!	Pending

4. SYSTEM DEVELOPMENT

4.1 INTRODUCTION

Systems development is the process of defining, designing, testing, and implementing a new software application or program.

4.2 PROCESS DESCRIPTION

- Task Identification: Begin by identifying all tasks and responsibilities that need to be accomplished. This could include work-related tasks, personal errands, appointments, and other commitments.
- Prioritization: Once tasks are identified, prioritize them based on urgency and importance. Use criteria such as deadlines, project dependencies, and impact to determine the order in which tasks should be completed.
- Task Entry: Record tasks in the to-do list application or tool of choice. Include relevant details such as task description, due date, priority level, and any additional notes or attachments.
- Organization: Organize tasks into categories or lists to facilitate management and clarity. Common categories may include work projects, personal tasks, errands, or recurring responsibilities.
- Scheduling: Allocate specific time slots or deadlines for completing tasks. Consider factors such as workload, available resources, and personal preferences when scheduling tasks.
- Execution: Begin working on tasks according to the prioritized list and schedule. Focus on completing high-priority tasks first and allocate time effectively to ensure progress on all tasks.
- Monitoring: Regularly review the to-do list to track progress and update task status. Mark tasks as completed, reschedule unfinished tasks if necessary, and adjust priorities as needed.

4.3 CODING

Index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8" />
<meta http-equiv="X-UA-Compatible" content="IE=edge" />
<meta name="viewport" content="width=device-width, initial-scale=1.0" />
<meta
name="description"
content="Stay organized with our user-friendly Calendar featuring
events, reminders, and a customizable interface. Built with HTML, CSS, and
JavaScript. Start scheduling today!"
/>
<meta
name="keywords"
content="calendar, events, reminders, javascript, html, css, "
/>
<link
rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.2.0/css/all.min.css"
integrity="sha512-
xh60/CkQoPOWdYTDqerDPCVd1SpvCA9XXcUnZS2FmJNp1coAFzvtCN9BmamE+4aHK8yyUHUSCcJHg
XloTyT2A=="
crossorigin="anonymous"
referrerpolicy="no-referrer"
/>
<link rel="stylesheet" href="style.css" />
<title>Calendar with Events</title>
</head>
<body>
<div class="container">
<div class="left">
<div class="calendar">
<div class="month">
<i class="fas fa-angle-left prev"></i>
<div class="date">december 2015</div>
<i class="fas fa-angle-right next"></i>
</div>
<div class="weekdays">
<div>Sun</div>
<div>Mon</div>
<div>Tue</div>
<div>Wed</div>
<div>Thu</div>
<div>Fri</div>
```

```

    <div>Sat</div>
  </div>
</div class="days"></div>
<div class="goto-today">
<div class="goto">
    <input type="text" placeholder="mm/yyyy" class="date-input" />
<button class="goto-btn">Go</button>
</div>
<button class="today-btn">Today</button>
</div>
</div>
</div>
<div class="right">
<div class="today-date">
<div class="event-day">wed</div>
<div class="event-date">12th december 2022</div>
</div>
<div class="events"></div>
<div class="add-event-wrapper">
<div class="add-event-header">
<div class="title">Add Event</div>
<i class="fas fa-times close"></i>
</div>
<div class="add-event-body">
<div class="add-event-input">
    <input type="text" placeholder="Event Name" class="event-name"
  />
</div>
<div class="add-event-input">
<input
type="text"
placeholder="Event Time From"
class="event-time-from"
/>
</div>
<div class="add-event-input">
<input
type="text"
placeholder="Event Time To"
class="event-time-to"
/>
</div>
</div>
<div class="add-event-footer">
<button class="add-event-btn">Add Event</button>
</div>
</div>
</div>

```

```

    <button class="add-event">
    <i class="fas fa-plus"></i>
    </button>
  </div>

  <div class="credits">
  <p>
  Done by
  <target="_blank">Trish</a>
  </p>
  </div>

    <script src="script.js"></script>
  </body>
</html>

```

Javascript.js

```

const calendar = document.querySelector(".calendar"),
date = document.querySelector(".date"),
daysContainer = document.querySelector(".days"),
prev = document.querySelector(".prev"),
next = document.querySelector(".next"),
todayBtn = document.querySelector(".today-btn"),
gotoBtn = document.querySelector(".goto-btn"),
dateInput = document.querySelector(".date-input"),
eventDay = document.querySelector(".event-day"),
eventDate = document.querySelector(".event-date"),
eventsContainer = document.querySelector(".events"),
addEventBtn = document.querySelector(".add-event"),
  addEventWrapper = document.querySelector(".add-event-wrapper "),
addEventCloseBtn = document.querySelector(".close "),
addEventTitle = document.querySelector(".event-name "),
addEventFrom = document.querySelector(".event-time-from "),
addEventTo = document.querySelector(".event-time-to "),
addEventSubmit = document.querySelector(".add-event-btn ");

let today = new Date();
let activeDay;
let month = today.getMonth();
let year = today.getFullYear();

const months = [
  "January",
  "February",
  "March",
  "April",
  "May",

```

```

    "June",
    "July",
    "August",
    "September",
    "October",
    "November",
    "December",
  ];

  // const eventsArr = [
  // {
  // day: 13,
  // month: 11,
  // year: 2022,
  // events: [
  // {
  // title: "Event 1 lorem ipsum dolar sit genfa tersd dsad ",
  // time: "10:00 AM",
  // },
  // {
  // title: "Event 2",
  // time: "11:00 AM",
  // },
  // ],
  // },
  // ];

const eventsArr = [];
getEvents();
console.log(eventsArr);

//function to add days in days with class day and prev-date next-date on
previous month and next month days and active on today
function initCalendar() {
const firstDay = new Date(year, month, 1);
const lastDay = new Date(year, month + 1, 0);
const prevLastDay = new Date(year, month, 0);
const prevDays = prevLastDay.getDate();
const lastDate = lastDay.getDate();
const day = firstDay.getDay();
const nextDays = 7 - lastDay.getDay() - 1;

date.innerHTML = months[month] + " " + year;

let days = "";

for (let x = day; x > 0; x--) {
days += `<div class="day prev-date">${prevDays - x + 1}</div>`;

```

```

    }

    for (let i = 1; i <= lastDate; i++) {
        //check if event is present on that day
        let event = false;
        eventsArr.forEach((eventObj) => {
            if (
                eventObj.day === i &&
                eventObj.month === month + 1 &&
                eventObj.year === year
            ) {
                event = true;
            }
        });
        if (
            i === new Date().getDate() &&
            year === new Date().getFullYear() &&
            month === new Date().getMonth()
        ) {
            activeDay = i;
            getActiveDay(i);
            updateEvents(i);
            if (event) {
                days += `<div class="day today active event">${i}</div>`;
            } else {
                days += `<div class="day today active">${i}</div>`;
            }
        } else {
            if (event) {
                days += `<div class="day event">${i}</div>`;
            } else {
                days += `<div class="day ">${i}</div>`;
            }
        }
    }

    for (let j = 1; j <= nextDays; j++) {
        days += `<div class="day next-date">${j}</div>`;
    }
    daysContainer.innerHTML = days;
    addListner();
}

//function to add month and year on prev and next button
function prevMonth() {
    month--;
    if (month < 0) {
        month = 11;
    }
}

```

```

    year--;
  }
  initCalendar();
}

function nextMonth() {
  month++;
  if (month > 11) {
    month = 0;
    year++;
  }
  initCalendar();
}

prev.addEventListener("click", prevMonth);
next.addEventListener("click", nextMonth);

initCalendar();

//function to add active on day
function addListner() {
  const days = document.querySelectorAll(".day");
  days.forEach((day) => {
    day.addEventListener("click", (e) => {
      getActiveDay(e.target.innerHTML);
      updateEvents(Number(e.target.innerHTML));
      activeDay = Number(e.target.innerHTML);
      //remove active
      days.forEach((day) => {
        day.classList.remove("active");
      });
      //if clicked prev-date or next-date switch to that month
      if (e.target.classList.contains("prev-date")) {
        prevMonth();
        //add active to clicked day afte month is change
        setTimeout(() => {
          //add active where no prev-date or next-date
          const days = document.querySelectorAll(".day");
          days.forEach((day) => {
            if (
              !day.classList.contains("prev-date") &&
              day.innerHTML === e.target.innerHTML
            ) {
              day.classList.add("active");
            }
          });
        }, 100);
      } else if (e.target.classList.contains("next-date")) {

```

```

    nextMonth();
    //add active to clicked day afte month is changed
    setTimeout(() => {
    const days = document.querySelectorAll(".day");
    days.forEach((day) => {
    if (
    !day.classList.contains("next-date") &&
    day.innerHTML === e.target.innerHTML
    ) {
    day.classList.add("active");
    }
    });
    }, 100);
    } else {
    e.target.classList.add("active");
    }
    });
    });
    }

    todayBtn.addEventListener("click", () => {
    today = new Date();
    month = today.getMonth();
    year = today.getFullYear();
    initCalendar();
    });

    dateInput.addEventListener("input", (e) => {
    dateInput.value = dateInput.value.replace(/^[0-9]/g, "");
    if (dateInput.value.length === 2) {
    dateInput.value += "/";
    }
    if (dateInput.value.length > 7) {
    dateInput.value = dateInput.value.slice(0, 7);
    }
    if (e.inputType === "deleteContentBackward") {
    if (dateInput.value.length === 3) {
    dateInput.value = dateInput.value.slice(0, 2);
    }
    }
    });

    gotoBtn.addEventListener("click", gotoDate);

    function gotoDate() {
    console.log("here");
    const dateArr = dateInput.value.split("/");
    if (dateArr.length === 2) {

```

```

    if (dateArr[0] > 0 && dateArr[0] < 13 && dateArr[1].length === 4) {
    month = dateArr[0] - 1;
    year = dateArr[1];
    initCalendar();
    return;
    }
    }
    alert("Invalid Date");
    }

    //function get active day day name and date and update eventday eventdate
    function getActiveDay(date) {
    const day = new Date(year, month, date);
    const dayName = day.toString().split(" ")[0];
    eventDay.innerHTML = dayName;
    eventDate.innerHTML = date + " " + months[month] + " " + year;
    }

    //function update events when a day is active
    function updateEvents(date) {
    let events = "";
    eventsArr.forEach((event) => {
    if (
    date === event.day &&
    month + 1 === event.month &&
    year === event.year
    ) {
    event.events.forEach((event) => {
    events += `<div class="event">
    <div class="title">
    <i class="fas fa-circle"></i>
    <h3 class="event-title">${event.title}</h3>
    </div>
    <div class="event-time">
    <span class="event-time">${event.time}</span>
    </div>
    </div>`;
    });
    }
    });
    if (events === "") {
    events = `<div class="no-event">
    <h3>No Events</h3>
    </div>`;
    }
    eventsContainer.innerHTML = events;
    saveEvents();
    }

```

```

//function to add event
addEventBtn.addEventListener("click", () => {
addEventWrapper.classList.toggle("active");
});

addEventCloseBtn.addEventListener("click", () => {
addEventWrapper.classList.remove("active");
});

document.addEventListener("click", (e) => {
  if (e.target !== addEventBtn && !addEventWrapper.contains(e.target)) {
addEventWrapper.classList.remove("active");
}
});

//allow 50 chars in eventtitle
addEventTitle.addEventListener("input", (e) => {
addEventTitle.value = addEventTitle.value.slice(0, 60);
});

function defineProperty() {
var osccred = document.createElement("div");
osccred.style.position = "absolute";
osccred.style.bottom = "0";
osccred.style.right = "0";
osccred.style.fontSize = "10px";
osccred.style.color = "#ccc";
osccred.style.fontFamily = "sans-serif";
osccred.style.padding = "5px";
osccred.style.background = "#fff";
osccred.style.borderTopLeftRadius = "5px";
osccred.style.borderBottomRightRadius = "5px";
osccred.style.boxShadow = "0 0 5px #ccc";
document.body.appendChild(osccred);
}

defineProperty();

//allow only time in eventtime from and to
addEventFrom.addEventListener("input", (e) => {
addEventFrom.value = addEventFrom.value.replace(/^[^0-9:]/g, "");
if (addEventFrom.value.length === 2) {
addEventFrom.value += ":";
}
if (addEventFrom.value.length > 5) {
addEventFrom.value = addEventFrom.value.slice(0, 5);
}
}

```

```

});

addEventTo.addEventListener("input", (e) => {
addEventTo.value = addEventTo.value.replace(/^[^0-9:]/g, "");
if (addEventTo.value.length === 2) {
addEventTo.value += ":";
}
if (addEventTo.value.length > 5) {
addEventTo.value = addEventTo.value.slice(0, 5);
}
});

//function to add event to eventsArr
addEventSubmit.addEventListener("click", () => {
const eventTitle = addEventTitle.value;
const eventTimeFrom = addEventFrom.value;
const eventTimeTo = addEventTo.value;
  if (eventTitle === "" || eventTimeFrom === "" || eventTimeTo === "") {
alert("Please fill all the fields");
return;
}

//check correct time format 24 hour
const timeFromArr = eventTimeFrom.split(":");
const timeToArr = eventTimeTo.split(":");
if (
timeFromArr.length !== 2 ||
timeToArr.length !== 2 ||
timeFromArr[0] > 23 ||
timeFromArr[1] > 59 ||
timeToArr[0] > 23 ||
timeToArr[1] > 59
) {
alert("Invalid Time Format");
return;
}

const timeFrom = convertTime(eventTimeFrom);
const timeTo = convertTime(eventTimeTo);

//check if event is already added
let eventExist = false;
eventsArr.forEach((event) => {
if (
event.day === activeDay &&
event.month === month + 1 &&
event.year === year
) {

```

```

    event.events.forEach((event) => {
    if (event.title === eventTitle) {
    eventExist = true;
    }
    });
    }
    });
    if (eventExist) {
    alert("Event already added");
    return;
    }
    const newEvent = {
    title: eventTitle,
    time: timeFrom + " - " + timeTo,
    };
    console.log(newEvent);
    console.log(activeDay);
    let eventAdded = false;
    if (eventsArr.length > 0) {
    eventsArr.forEach((item) => {
    if (
    item.day === activeDay &&
    item.month === month + 1 &&
    item.year === year
    ) {
    item.events.push(newEvent);
    eventAdded = true;
    }
    });
    }

    if (!eventAdded) {
    eventsArr.push({
    day: activeDay,
    month: month + 1,
    year: year,
    events: [newEvent],
    });
    }

    console.log(eventsArr);
    addEventWrapper.classList.remove("active");
    addEventTitle.value = "";
    addEventFrom.value = "";
    addEventTo.value = "";
    updateEvents(activeDay);
    //select active day and add event class if not added
    const activeDayEl = document.querySelector(".day.active");

```

```

    if (!activeDayEl.classList.contains("event")) {
    activeDayEl.classList.add("event");
    }
    });

    //function to delete event when clicked on event
    eventsContainer.addEventListener("click", (e) => {
    if (e.target.classList.contains("event")) {
    if (confirm("Are you sure you want to delete this event?")) {
    const eventTitle = e.target.children[0].children[1].innerHTML;
    eventsArr.forEach((event) => {
    if (
    event.day === activeDay &&
    event.month === month + 1 &&
    event.year === year
    ) {
    event.events.forEach((item, index) => {
    if (item.title === eventTitle) {
    event.events.splice(index, 1);
    }
    });
    //if no events left in a day then remove that day from eventsArr
    if (event.events.length === 0) {
    eventsArr.splice(eventsArr.indexOf(event), 1);
    //remove event class from day
    const activeDayEl = document.querySelector(".day.active");
    if (activeDayEl.classList.contains("event")) {
    activeDayEl.classList.remove("event");
    }
    }
    });
    updateEvents(activeDay);
    }
    });

    //function to save events in local storage
    function saveEvents() {
    localStorage.setItem("events", JSON.stringify(eventsArr));
    }

    //function to get events from local storage
    function getEvents() {
    //check if events are already saved in local storage then return event else
    nothing
    if (localStorage.getItem("events") === null) {
    return;

```

```

    }
    eventsArr.push(...JSON.parse(localStorage.getItem("events")));
  }

function convertTime(time) {
  //convert time to 24 hour format
  let timeArr = time.split(":");
  let timeHour = timeArr[0];
  let timeMin = timeArr[1];
  let timeFormat = timeHour >= 12 ? "PM" : "AM";
  timeHour = timeHour % 12 || 12;
  time = timeHour + ":" + timeMin + " " + timeFormat;
  return time;
}

```

Style.css

```

:root {
  --primary-clr: #b38add;
}
* {
  margin: 0;
  padding: 0;
  box-sizing: border-box;
  font-family: "Poppins", sans-serif;
}
/* nice scroll bar */
::-webkit-scrollbar {
  width: 5px;
}
::-webkit-scrollbar-track {
  background: #f5f5f5;
  border-radius: 50px;
}
::-webkit-scrollbar-thumb {
  background: var(--primary-clr);
  border-radius: 50px;
}

body {
  position: relative;
  min-height: 100vh;
  display: flex;
  align-items: center;
  justify-content: center;
  padding-bottom: 30px;
  background-color: #e2e1dc;
}

```

```
.container {
position: relative;
width: 1200px;
min-height: 850px;
margin: 0 auto;
padding: 5px;
color: #fff;
display: flex;

border-radius: 10px;
background-color: #373c4f;
}
.left {
width: 60%;
padding: 20px;
}
.calendar {
position: relative;
width: 100%;
height: 100%;
display: flex;
flex-direction: column;
flex-wrap: wrap;
justify-content: space-between;
color: #878895;
border-radius: 5px;
background-color: #fff;
}
/* set after behind the main element */
.calendar::before,
.calendar::after {
content: "";
position: absolute;
top: 50%;
left: 100%;
width: 12px;
height: 97%;
border-radius: 0 5px 5px 0;
background-color: #d3d4d6d7;
transform: translateY(-50%);
}
.calendar::before {
height: 94%;
left: calc(100% + 12px);
background-color: rgb(153, 153, 153);
}
.calendar .month {
width: 100%;
```

```

    height: 150px;
display: flex;
align-items: center;
    justify-content: space-between;
padding: 0 50px;
font-size: 1.2rem;
font-weight: 500;
    text-transform: capitalize;
}
.calendar .month .prev,
.calendar .month .next {
cursor: pointer;
}
.calendar .month .prev:hover,
.calendar .month .next:hover {
color: var(--primary-clr);
}
.calendar .weekdays {
width: 100%;
height: 100px;
display: flex;
align-items: center;
    justify-content: space-between;
padding: 0 20px;
font-size: 1rem;
font-weight: 500;
    text-transform: capitalize;
}
.weekdays div {
width: 14.28%;
height: 100%;
display: flex;
align-items: center;
justify-content: center;
}
.calendar .days {
width: 100%;
display: flex;
flex-wrap: wrap;
    justify-content: space-between;
padding: 0 20px;
font-size: 1rem;
font-weight: 500;
margin-bottom: 20px;
}
.calendar .days .day {
width: 14.28%;
height: 90px;

```

```

    display: flex;
    align-items: center;
    justify-content: center;
    cursor: pointer;
    color: var(--primary-clr);
    border: 1px solid #f5f5f5;
}
.calendar .days .day:nth-child(7n + 1) {
border-left: 2px solid #f5f5f5;
}
.calendar .days .day:nth-child(7n) {
border-right: 2px solid #f5f5f5;
}
.calendar .days .day:nth-child(-n + 7) {
border-top: 2px solid #f5f5f5;
}
.calendar .days .day:nth-child(n + 29) {
border-bottom: 2px solid #f5f5f5;
}

    .calendar .days .day:not(.prev-date, .next-date):hover {
color: #fff;
background-color: var(--primary-clr);
}
.calendar .days .prev-date,
.calendar .days .next-date {
color: #b3b3b3;
}
.calendar .days .active {
position: relative;
font-size: 2rem;
color: #fff;
background-color: var(--primary-clr);
}
.calendar .days .active::before {
content: "";
position: absolute;
top: 0;
left: 0;
width: 100%;
height: 100%;
box-shadow: 0 0 10px 2px var(--primary-clr);
}
.calendar .days .today {
font-size: 2rem;
}
.calendar .days .event {
position: relative;

```

```

}
.calendar .days .event::after {
content: "";
position: absolute;
bottom: 10%;
left: 50%;
width: 75%;
height: 6px;
border-radius: 30px;
transform: translateX(-50%);
background-color: var(--primary-clr);
}
.calendar .days .day:hover.event::after {
background-color: #fff;
}
.calendar .days .active.event::after {
background-color: #fff;
bottom: 20%;
}
.calendar .days .active.event {
padding-bottom: 10px;
}
.calendar .goto-today {
width: 100%;
height: 50px;
display: flex;
align-items: center;
justify-content: space-between;
gap: 5px;
padding: 0 20px;
margin-bottom: 20px;
color: var(--primary-clr);
}
.calendar .goto-today .goto {
display: flex;
align-items: center;
border-radius: 5px;
overflow: hidden;
border: 1px solid var(--primary-clr);
}
.calendar .goto-today .goto input {
width: 100%;
height: 30px;
outline: none;
border: none;
border-radius: 5px;
padding: 0 20px;
color: var(--primary-clr);
}

```

```

border-radius: 5px;
}
.calendar .goto-today button {
padding: 5px 10px;
border: 1px solid var(--primary-clr);
border-radius: 5px;
background-color: transparent;
cursor: pointer;
color: var(--primary-clr);
}
.calendar .goto-today button:hover {
color: #fff;
background-color: var(--primary-clr);
}
.calendar .goto-today .goto button {
border: none;
border-left: 1px solid var(--primary-clr);
border-radius: 0;
}
.container .right {
position: relative;
width: 40%;
min-height: 100%;
padding: 20px 0;
}

.right .today-date {
width: 100%;
height: 50px;
display: flex;
flex-wrap: wrap;
gap: 10px;
align-items: center;
justify-content: space-between;
padding: 0 40px;
padding-left: 70px;
margin-top: 50px;
margin-bottom: 20px;
text-transform: capitalize;
}
.right .today-date .event-day {
font-size: 2rem;
font-weight: 500;
}
.right .today-date .event-date {
font-size: 1rem;
font-weight: 400;
color: #878895;
}

```

```

}
.events {
width: 100%;
height: 100%;
max-height: 600px;
overflow-x: hidden;
overflow-y: auto;
display: flex;
flex-direction: column;
padding-left: 4px;
}
.events .event {
position: relative;
width: 95%;
min-height: 70px;
display: flex;
justify-content: center;
flex-direction: column;
gap: 5px;
padding: 0 20px;
padding-left: 50px;
color: #fff;
background: linear-gradient(90deg, #3f4458, transparent);
cursor: pointer;
}
/* even event */
.events .event:nth-child(even) {
background: transparent;
}
.events .event:hover {
background: linear-gradient(90deg, var(--primary-clr), transparent);
}
.events .event .title {
display: flex;
align-items: center;
pointer-events: none;
}
.events .event .title .event-title {
font-size: 1rem;
font-weight: 400;
margin-left: 20px;
}
.events .event i {
color: var(--primary-clr);
font-size: 0.5rem;
}
.events .event:hover i {
color: #fff;
}

```

```

}
.events .event .event-time {
font-size: 0.8rem;
font-weight: 400;
color: #878895;
margin-left: 15px;
pointer-events: none;
}

.events .event:hover .event-time {
color: #fff;
}
/* add tick in event after */
.events .event::after {
content: "✓";
position: absolute;
top: 50%;
right: 0;
font-size: 3rem;
line-height: 1;
display: none;
align-items: center;
justify-content: center;
opacity: 0.3;
color: var(--primary-clr);
transform: translateY(-50%);
}
.events .event:hover::after {
display: flex;
}
.add-event {
position: absolute;
bottom: 30px;
right: 30px;
width: 40px;
height: 40px;
display: flex;
align-items: center;
justify-content: center;
font-size: 1rem;
color: #878895;
border: 2px solid #878895;
opacity: 0.5;
border-radius: 50%;
background-color: transparent;
cursor: pointer;
}
.add-event:hover {
opacity: 1;
}

```

```

}
.add-event i {
pointer-events: none;
}
.events .no-event {
width: 100%;
height: 100%;
display: flex;
align-items: center;
justify-content: center;
font-size: 1.5rem;
font-weight: 500;
color: #878895;
}
.add-event-wrapper {
position: absolute;
bottom: 100px;
left: 50%;
width: 90%;
max-height: 0;
overflow: hidden;
border-radius: 5px;
background-color: #fff;
transform: translateX(-50%);
transition: max-height 0.5s ease;
}
.add-event-wrapper.active {
max-height: 300px;
}
.add-event-header {
width: 100%;
height: 50px;
display: flex;
align-items: center;
justify-content: space-between;
padding: 0 20px;
color: #373c4f;
border-bottom: 1px solid #f5f5f5;
}
.add-event-header .close {
font-size: 1.5rem;
cursor: pointer;
}
.add-event-header .close:hover {
color: var(--primary-clr);
}
.add-event-header .title {
font-size: 1.2rem;

```

```

    font-weight: 500;
}
.add-event-body {
width: 100%;
height: 100%;
display: flex;
flex-direction: column;
gap: 5px;
padding: 20px;
}
.add-event-body .add-event-input {
width: 100%;
height: 40px;
display: flex;
align-items: center;
justify-content: space-between;
gap: 10px;
}
.add-event-body .add-event-input input {
width: 100%;
height: 100%;
outline: none;
border: none;
border-bottom: 1px solid #f5f5f5;
padding: 0 10px;
font-size: 1rem;
font-weight: 400;
color: #373c4f;
}
.add-event-body .add-event-input input::placeholder {
color: #a5a5a5;
}
.add-event-body .add-event-input input:focus {
border-bottom: 1px solid var(--primary-clr);
}
.add-event-body .add-event-input input:focus::placeholder {
color: var(--primary-clr);
}
.add-event-footer {
display: flex;
align-items: center;
justify-content: center;
padding: 20px;
}
.add-event-footer .add-event-btn {
height: 40px;
font-size: 1rem;
font-weight: 500;

```

```

    outline: none;
border: none;
color: #fff;
background-color: var(--primary-clr);
border-radius: 5px;
cursor: pointer;
padding: 5px 10px;
border: 1px solid var(--primary-clr);
}
.add-event-footer .add-event-btn:hover {
background-color: transparent;
color: var(--primary-clr);
}

/* media queries */

@media screen and (max-width: 1000px) {
body {
align-items: flex-start;
justify-content: flex-start;
}
.container {
min-height: 100vh;
flex-direction: column;
border-radius: 0;
}
.container .left {
width: 100%;
height: 100%;
padding: 20px 0;
}
.container .right {
width: 100%;
height: 100%;
padding: 20px 0;
}
.calendar::before,
.calendar::after {
top: 100%;
left: 50%;
width: 97%;
height: 12px;
border-radius: 0 0 5px 5px;
transform: translateX(-50%);
}
.calendar::before {
width: 94%;
top: calc(100% + 12px);
}

```

```
}  
.events {  
padding-bottom: 340px;  
}  
.add-event-wrapper {  
bottom: 100px;  
}  
}  
@media screen and (max-width: 500px) {  
.calendar .month {  
height: 75px;  
}  
.calendar .weekdays {  
height: 50px;  
}  
.calendar .days .day {  
height: 40px;  
font-size: 0.8rem;  
}  
.calendar .days .day.active,  
.calendar .days .day.today {  
font-size: 1rem;  
}  
.right .today-date {  
padding: 20px;  
}  
}  
  
.credits {  
position: absolute;  
bottom: 0;  
left: 0;  
right: 0;  
text-align: center;  
padding: 10px;  
font-size: 12px;  
color: #fff;  
background-color: #b38add;  
}  
.credits a {  
color: #fff;  
text-decoration: none;  
font-weight: 600;  
}  
.credits a:hover {  
text-decoration: underline;  
}
```

README.md

```
"# Responsive-Calendar-with-Events"
```

5. SYSTEM TESTING AND IMPLEMENTATION

1. INTRODUCTION

System testing is defined as the testing of a complete and fully integrated software product. This testing falls in black box testing wherein knowledge of the design of the code is not a pre-requisite and is done by the testing team. System testing tests the design and behavior of the system and the expectation of the user.

2. DEBUGGING

Debugging is the process of finding and fixing errors or bugs in the source code of any software. When software does not work as expected, computer programmers study the code to determine why any errors occurred.

5.2.1 UNIT TESTING



Task Creation: Test the functionality of creating a new task. Verify that when a user inputs task details such as title, description, due date, and priority, the task is created successfully and stored in the database.



Task Editing: Test the functionality of editing an existing task. Verify that when a user modifies task details such as title, description, due date, or priority, the changes are reflected correctly in the database without affecting other tasks.



Task Deletion: Test the functionality of deleting a task. Verify that when a user deletes a task, it is removed from the database and no longer appears in the list of tasks.



Task Prioritization: Test the functionality of prioritizing tasks. Verify that tasks are displayed in the correct order based on their priority level, with higher-priority tasks appearing before lower-priority tasks.



Task Categorization: Test the functionality of categorizing tasks into different lists or categories. Verify that tasks are assigned to the correct category and displayed accordingly.



Reminder Functionality: Test the functionality of setting reminders for tasks. Verify that reminders are triggered at the specified time or date, and that users receive notifications or alerts as expected.



B.VOC SOFTWARE DEVELOPMENT

Task Completion: Test the functionality of marking tasks as completed. Verify that when a user marks a task as completed, it is updated in the database and displayed accordingly in the list of tasks.

2. INTEGRATION TESTING

To-do list integration testing involves verifying the seamless interaction between different components of the system, such as task creation, categorization, reminders, and collaboration tools. It ensures that all integrated functionalities work together as expected, enabling smooth workflow and user experience across the application.

3. SCOPE FOR FUTURE ENHANCEMENT

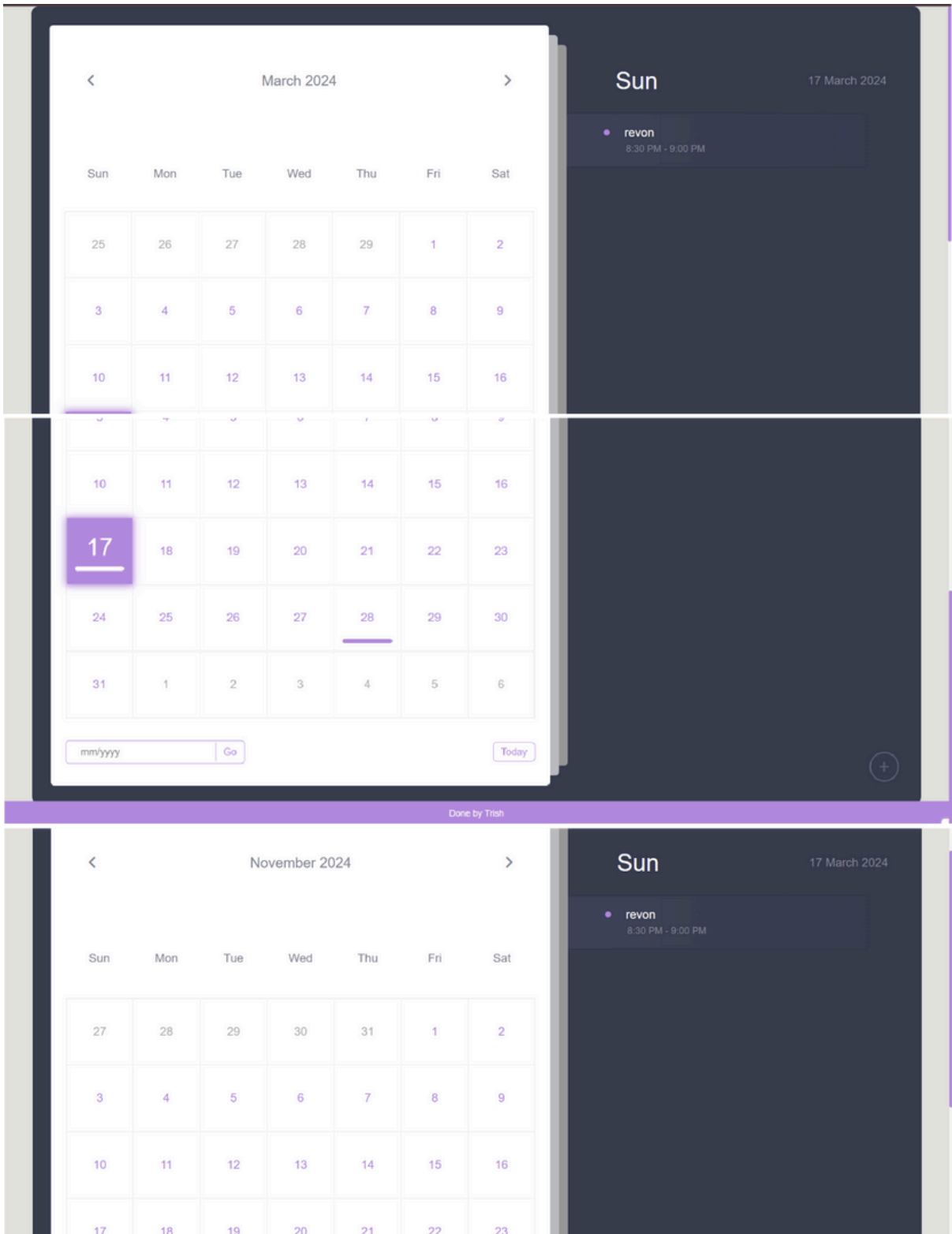
Future enhancements for the to-do list system may include incorporating artificial intelligence for personalized task recommendations, implementing advanced collaboration features like real-time task updates, integrating with smart home devices for voice-controlled task management, and expanding platform support to include wearable devices for on-the-go task management.

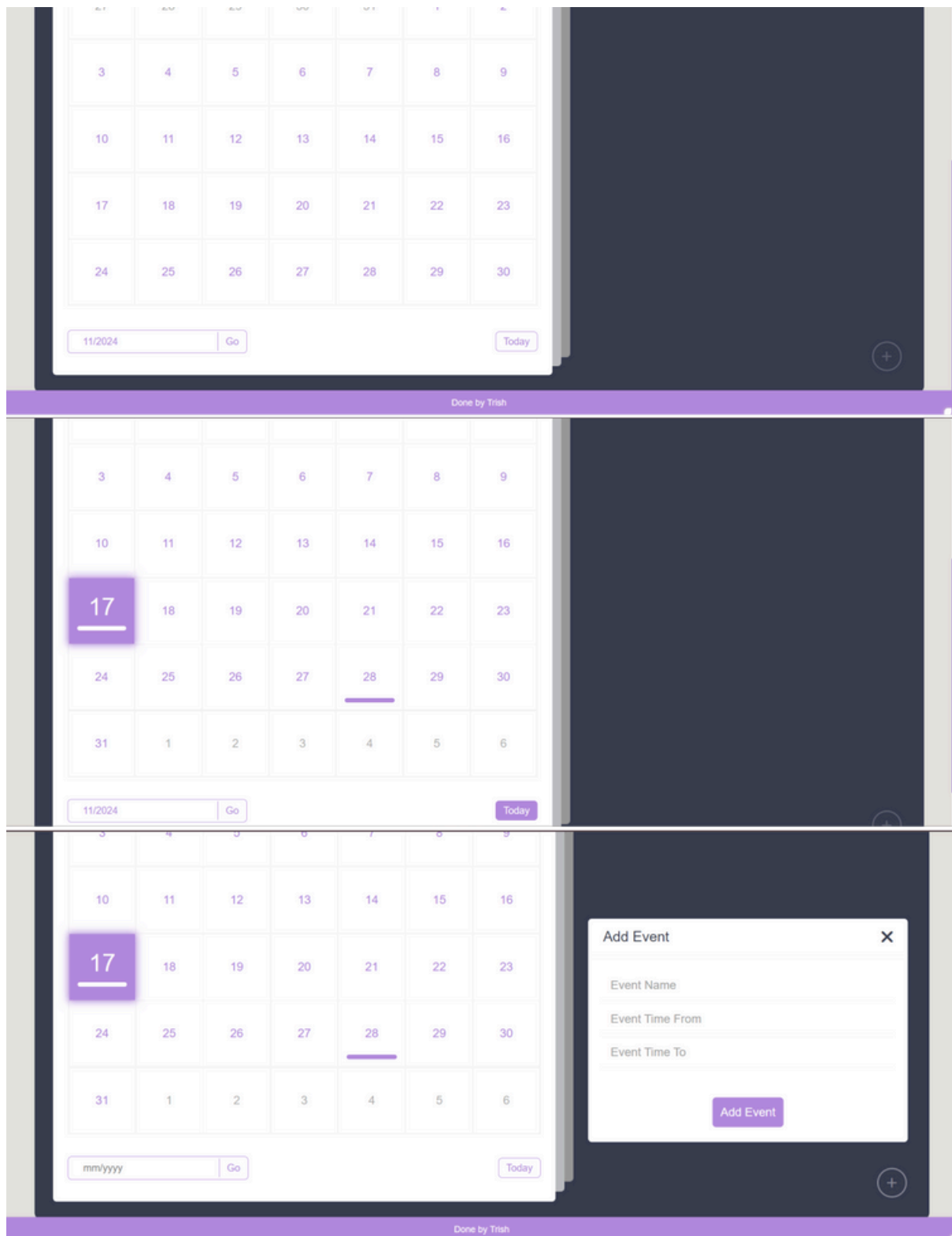
6. CONCLUSION

- In conclusion, it helps make the user freely without
- difficulty. Permission grants help to not miss
- important events.

View of the tasks that are completed

7. APPENDIX





8. BIBLIOGRAPHY

Site referred

- <https://play.google.com/store/apps/details?id=com.splendapps.splend>
o
- <https://play.google.com/store/apps/details?id=todolist.scheduleplanner.dailyplanner.todo.reminders>
- <https://docplayer.net/13534020-Software-requirements-specification-task-management-system-for-prepared-by-version-1-0-group-name-pink-and-purple-date.html>
- <https://pdfs.semanticscholar.org/8a08/7a0c6ffbd3b2c249f652dd930be8ff58568f.pdf>
- <https://youtu.be/6EVgmpm4z5U?si=hlqkRR83a9v54mo8>