FINAL SEMESTER ASSESSMENT (FSA) B.TECH. (CSE) VI SEMESTER

UE18CS355 – OBJECT ORIENTED ANALYSIS AND DESIGN WITH SOFTWARE ENGINEERING LABORATORY

PROJECT REPORT

ON

CALENDAR MANAGEMENT SYSTEM

SUBMITTED BY

NAME	SRN
1) Vishesh P	PES1201800314
2) Samarth G Vasist	PES1201801432
3) Navaneeth M	PES1201801597

JANUARY – MAY 2021

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

RR CAMPUS,
BENGALURU – 560100, KARNATAKA,
INDIA

TABLE OF CONTENTS

ABSTRACT

In today's world everyone in the world has plans and agendas. People will need to plan new events in the future and keep track of their events. Fortunately, there exists a tool that brings all these requirements under one umbrella and meets them, namely the Calendar. In our project, we have attempted to build a Calendar with basic functionalities such as Viewing the Calendar, Setting/Deleting Events and finally sending reminders for events via Email.

The user will first need to signup on the application following which an account will be created in his/her name. Then the user may sign into their account using their credentials, upon which the user will be provided three options : Views, add event and delete event. The Views module mainly displays the calendar for the day/week/month and also provides the user all the events scheduled for that particular day, week and month respectively. The add event module is used to add an event to the list of events which is unique to each user. Additionally, the add event module allows the user to include an optional reminder for each event, such that when the time is approximately equal to the event's start time the user will receive an email notification regarding that particular event. The delete event module is used to delete an event from the corresponding user's list.

After noting down the requirements specified in the SRS document and the Project Plan, we went about designing the High-level, Class, Activity, Sequence and State diagrams for the use cases mentioned. We then went on to implement our project using Python-Flask combined with Sqlite as our backend and with HTML/CSS for the User Interface. The description of each use case implemented has been provided in this document along with the test cases for each use case.

1. SOFTWARE REQUIREMENTS SPECIFICATION

1.1 Introduction

1.1.1 Purpose

This SRS aims to describe the hosted web application to be built for release v1.0 of the Calendar Management Application. The document describes the scope of the product, which is to provide a calendar application as a hosted and managed service.

1.1.2 Intended Audience

1.1.2.1 Software Developers

Developers will analyse, design, implement and integrate modules of the application. Developers will ensure that the product being designed is in compliance with this SRS document in terms of both functional and nonfunctional requirements.

1.1.2.2 Software Testers

Testers will ensure the integrity and compliance of modules before final deployments. Testers will also look into security aspects of the product and ensure integrity of the same.

1.1.3 Product Scope

The scope of this project is to take to completion the development of a calendar management application provided as a service. The application will provide a general purpose calendar and will additionally allow users to create events and keep track of them through scheduled reminders.

1.2 Overall Description

1.2.1 Product Perspective

The product provides a calendar management app as a hosted service. It works as an alternate to already existing calendar applications but avoids the need to be installed as a user application. Users can access their respective calendars through a web interface. As the application is deployed on a hosted platform, Heroku in this case, users can access it through any device, be it a computer, smartphone, tablet, etc.

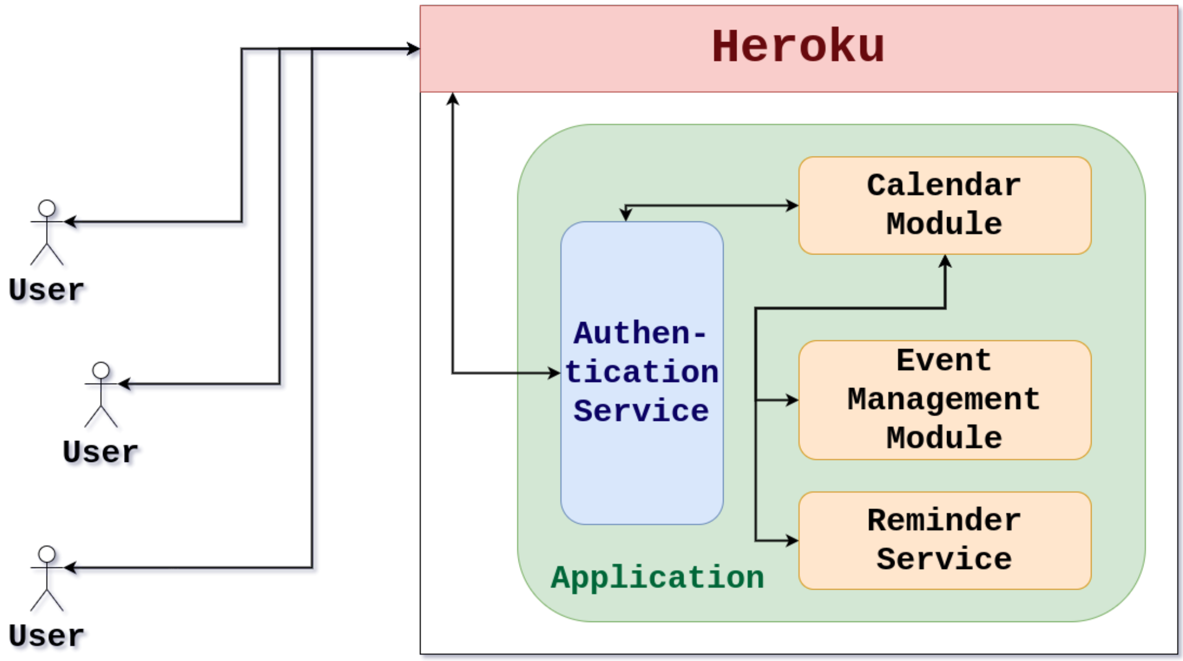


Fig 1.1

1.2.2 Product Functions

Major functionalities of the product are as follows:

- User friendly calendar UI
- Secure authentication for users
- Event Manager
- Event Reminder

1.2.3 User Classes and Characteristics

1.2.3.1 Students

This app will help students maintain a good checklist of events throughout their academic year along with assignment/project deadlines and exam schedules.

1.2.3.2 Working Professionals

Working professionals can leverage this application to schedule day to day events along with important meetings, deadlines, etc.

1.2.4 Operating Environment

The application will be hosted on Heroku, a Cloud Platform as a Service. This enables our application to be used through the web from any device with browser capabilities. Use of Heroku abstracts hardware requirements and helps focus purely on application development.

OS used will be a variant of one of many Linux Server distributions. Additional components will include a database, SQLite, React for frontend and Flask for backend.

1.2.5 Design and Implementation Constraints

While the application provides flexibility with regards to its features and their implementation, the application relies on Heroku's limit for the maximum number of users that can use the app. Security concerns arise during the process of user authentication and rely completely on the authentication service used.

1.2.6 Assumptions and Dependencies

- Use of a third party Authentication Service will reduce development time and effort.
- Alignment of UI components will not be an issue across different devices.
- Application will run the same irrespective of client device.

1.3 External Interface Requirements

1.3.1 User Interfaces

- The user is presented with a flexible grid layout running on the browser, that is two panes on landscape screens, and one pane on portrait screens. The smaller pane on the left contains contextual details, while the large main pane will be a grid, having timeslots. On portrait screens, the left pane will be accessible from a hamburger menu.
- The UI font will be a wide Sans typeface.
- The application will be hosted on Heroku.
- Red highlights occur around fields with invalid input.
- The interface will primarily use gray shades, with only a few colour highlights for the purposes of clear organisation and easy discoverability of important information.

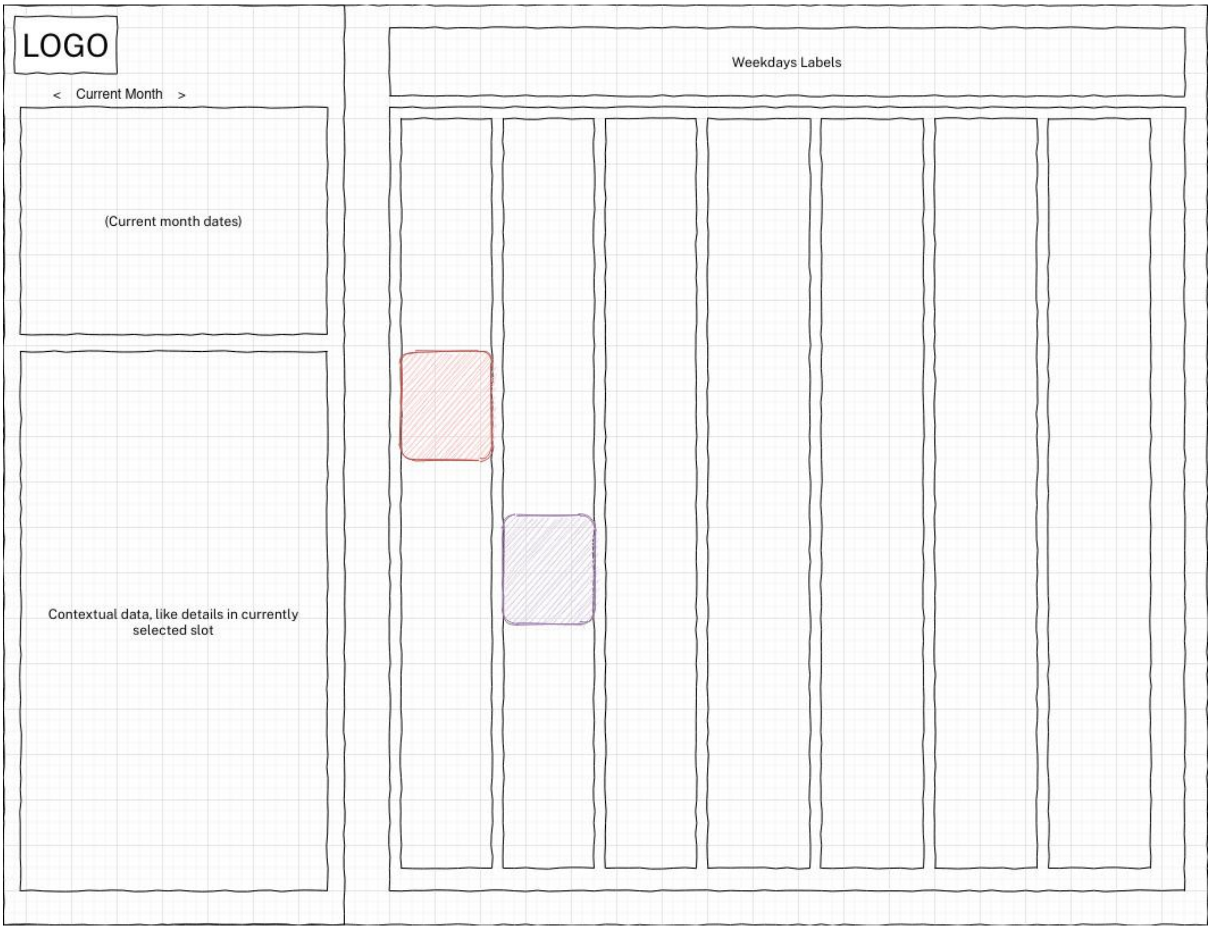


Fig 1.2

1.3.2 Software Interfaces

- We use SQLite (3.34) as our database to store events, reminders and appointments. It enables export and import with different accounts.
- UI and Frontend uses React for rendering the components on the client side.
- Backend implemented in Python using the Flask framework.
- Heroku's PaaS, for deploying the application.

1.3.3 Communications Interfaces

- Web browser with a working internet connection for accessing and viewing this app.
- Commands of registration, retrieval of events and reminders are handled through an HTTP requests-based API.

1.4 Analysis Models

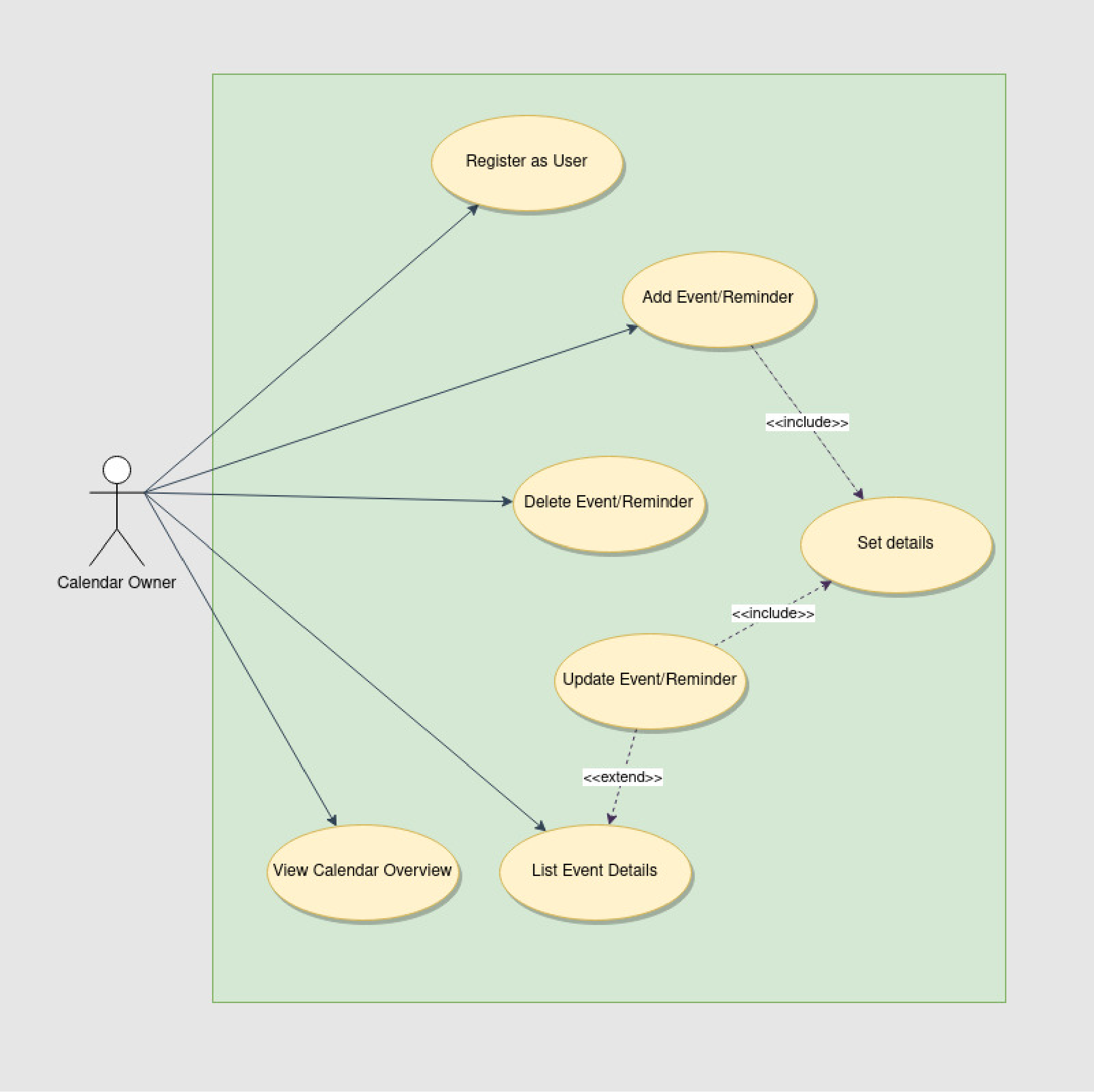


Fig 1.3 Use case diagram for Calendar Management.

1.5 System Features

1.5.1 Calendar

1.5.1.1 Description and Priority

Calendar is the visual representation of the Days and Months. Events and Reminders are reflected on the calendar.

This feature has High priority.

1.5.1.2 Stimulus/Response Sequences

A user creates a personal calendar upon creating an account through the login service.

1.5.1.3 Functional Requirements

REQ-1: Users should be able to see the events and reminders associated with their personal calendar

REQ-2: Users should be able to navigate to the next/previous months and weeks

REQ-3: Users should be allowed to display the different views of the calendar including week and month views

1.5.2 Events

1.5.2.1 Description and Priority

An event is a slice of time that the user wants to keep track of or have reminders for. This could be work related, personal events or national/international holidays.

This feature has High priority.

1.5.2.2 Stimulus/Response Sequences

A user can create an event through using the *Create Event* dialogue upon which the system will add the event to the users personal calendar.

1.5.2.3 Functional Requirements

REQ-1: Users should be able to create Events with an Event Title, Date and Time Period

REQ-2: Users should be allowed to associate Reminders with events (See section 5.2)

REQ-3: The event should be reflected in the calendar once created

REQ-4: The user should be able to edit or delete the event

1.5.3 Reminders

1.5.3.1 Description and Priority

A Reminder is a pop up notification that is associated with an Event that prompts the user of the events occurrence.

This feature has Medium priority.

1.5.3.2 Stimulus/Response Sequences

A user can create reminders through *Create Reminder* or *Create Event* dialogue which in turn sends a request to the system that prompts the user through a pop up notification when the reminder occurs.

1.5.3.3 Functional Requirements

REQ-1: Users should be able to create Reminders with a Reminder Title, Date and Time Period

REQ-2: Users should be able to choose the method of receiving the notification

REQ-3: The reminder should be reflected in the calendar once created

REQ-4: The user should be able to edit or delete the reminder

1.6 Other Nonfunctional Requirements

1.6.1 Performance Requirements

1. The system should support at least 200 concurrent users
2. Calendars must take no longer than one minute to be created
3. Events should be created and processed by the server in less than a minute
4. Reminders should be created and processed by the server in less than half a minute

1.6.2 Safety Requirements

1. The System does not disclose any personal information about the users
2. The Application does not grant Access to an unauthorized user
3. The Application shall not communicate with any other devices or servers while in use by the user

1.6.3 Security Requirements

1. Heroku provides Network, Data and System security. Since the app is hosted on Heroku, these advantages applies to the Calendar App as well
2. User Authentication is done by a third party service

1.6.4 Software Quality Attributes

1. The calendar should provide Availability to the user for access at any time
2. The app should also be Usable, implying the effort required to learn, operate, prepare input, and interpret output should be low

1.6.5 Business Rules

A calendar application that's hosted on Heroku's PaaS enables the user to access the Calendar from any device with a simple login. We also get security and availability services from this platform. Using React for the frontend and flask for the backend makes the codebase easy to maintain and develop.

1.7 Other Requirements

The Calendar Application uses a SQLite (3.34) as the database service for storing calendar events and reminder related information. This enables import and export of the Calendar along with reliability and consistency.

Appendix A: Field Layouts

An Excel sheet containing field layouts and properties/attributes and report requirements.

User registration

Field	Length	Data Type	Description	Is Mandatory
Email ID	255	String		Y
Password	255	String		Y

Events and Reminders

Event	Reminder
Event Name	Reminder Label
Date and Time	Time
Event Description	Frequency
Conference link	

Appendix B: Requirement Traceability Matrix

Sl. No	Requirement ID	Brief Description of Requirement	Architecture Reference	Design Reference	Code File Reference	Test Case ID	System Test Case ID
1	RT-01	Display the daily view of calendar	Fig 3.1, Fig 3.2, Fig 3.3, Fig 3.4	Fig 3.7, Fig 3.8	app.py, forms.py, dailyviews.html, layout1.html, main.css	UT-01	ST-10
2	RT-02	Display the weekly view of the calendar	Fig 3.1, Fig 3.2, Fig 3.3, Fig 3.4	Fig 3.7, Fig 3.8	app.py, forms.py, weeklyviews.html, layout1.html, main.css	UT-02	ST-10
3	RT-03	Display the monthly view on the calendar	Fig 3.1, Fig 3.2, Fig 3.3, Fig 3.4	Fig 3.7, Fig 3.8	app.py, forms.py, monthlyviews.html, layout1.html, main.css	UT-03	ST-10
4	RT-04	Viewing events under daily views	Fig 3.1, Fig 3.2, Fig 3.3, Fig 3.4	Fig 3.7, Fig 3.8	app.py, forms.py,dailyviews.html, layout1.html, main.css	IT-05	ST-10
5	RT-05	Viewing events under weekly views	Fig 3.1, Fig 3.2, Fig 3.3, Fig 3.4	Fig 3.7, Fig 3.8	app.py, forms.py, weeklyviews.html, layout1.html, main.css	IT-06	ST-10

6	RT-06	Viewing events under monthly views	Fig 3.1, Fig 3.2, Fig 3.3, Fig 3.4	Fig 3.7, Fig 3.8	app.py, forms.py, monthlyviews.html, layout1.html, main.css	IT-07	ST-10
7	RT-07	Add an event to be scheduled on a particular date	Fig 3.1, Fig 3.2, Fig 3.3, Fig 3.4	Fig 3.5, Fig 3.9, Fig 3.10	app.py, forms.py, new_event.html, layout1.html, main.css	UT-11	ST-10
8	RT-08	Delete an event which is already scheduled on particular day	Fig 3.1, Fig 3.2, Fig 3.3, Fig 3.4	Fig 3.5, Fig 3.9, Fig 3.10	app.py, forms.py, del_event.html, layout1.html, main.css	UT-14	ST-10
9	RT-09	Setting a reminder to an event	Fig 3.1, Fig 3.2, Fig 3.3, Fig 3.4	Fig 3.6, Fig 3.11, Fig 3.12	app.py, forms.py, new_event.html, layout1.html, main.css	IT-19	ST-10
10	RT-10	Sending Email Reminders	Fig 3.1, Fig 3.2, Fig 3.3, Fig 3.4	Fig 3.6 Fig 3.11, Fig 3.12	app.py, forms.py, login.html, layout1.html, main.css	UT-21	ST-10

2. Project Plan

2.1 Software Development Life Cycle Model

The model chosen for the Calendar Management Project is the Prototype Model.
The prototyping model is a method in which a prototype is created, tested and then reconstructed as necessary until an appropriate end product is attained.
The benefits of this model are as follows:

- The defects or bugs can be detected at an early stage
- It helps in reducing cost and time
- It helps in providing feedback from the user and changes the things accordingly
- it is easy to identify the missing, confusing and difficult functionalities

2.2 Tool-kits and Software Employed

2.2.1 Planning Tool

The project uses Trello as a planning tool for efficient collaboration and organisation. Trello provides boards and cards to arrange all requirements and makes checking progress of the project at any time easy to do.

2.2.2 Design Tool

Draw.io is used as the design tool for visual based diagrams that help map out various features and architecture of the project.

2.2.3 Version Control

Git is a distributed version control system that is used due to its Simplicity, Branch Workflow and Open-Source Collaboration based approach.

2.2.4 Development Tool

npm (node package manager) is used for creating the boiler-plate react code and also helps with managing third party packages used for the project's frontend.
Flask is a web framework used for structuring out the backend of the project and allows a quick and easy start to the application as well as the ability to scale up the application as needed.

2.2.5 Bug Tracking

We use GitHub issues for bug tracking and resolving. GitHub issues focusses on collaborative bug solving. Since the project will also be hosted on GitHub, this choice is convenient and effective.

2.2.6 Testing Tool

We use inbuilt frameworks for testing. The code written in python will be tested and linted using pytest and pylist respectively. We also use npm test for testing the React frontend.

2.3 Project Functionalities

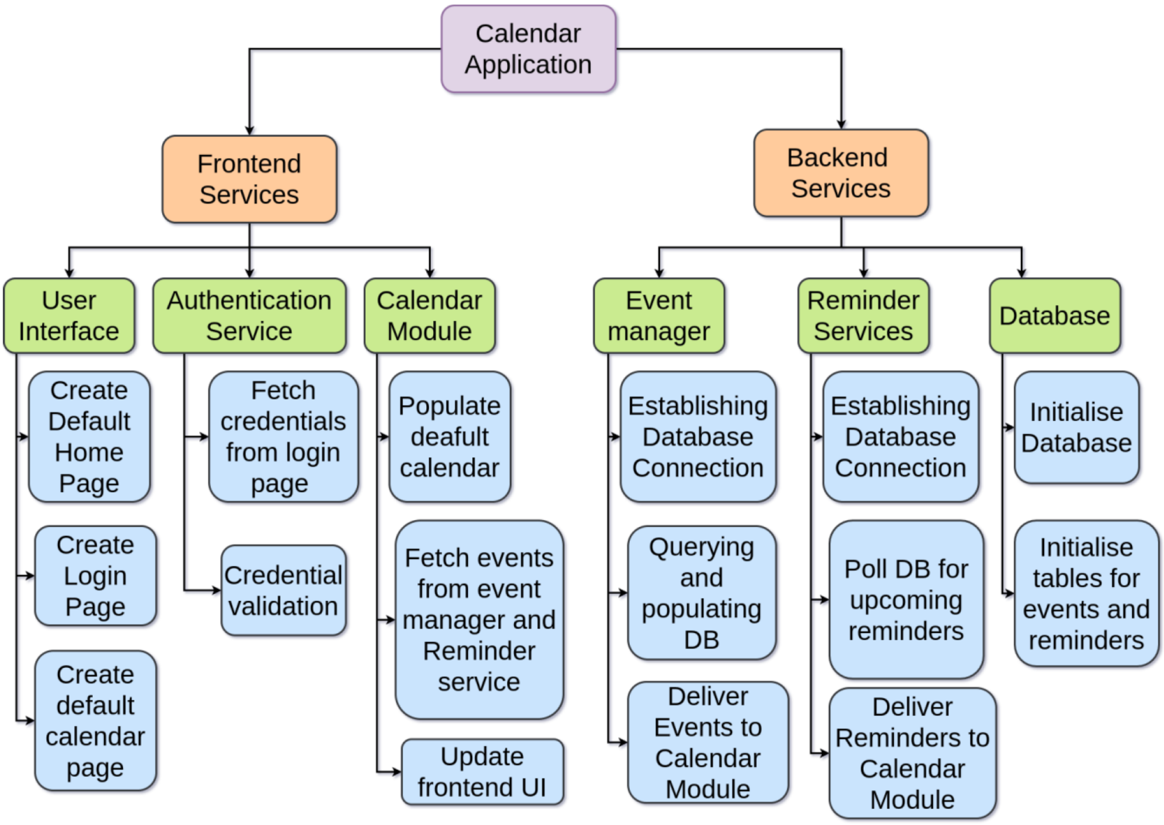

Calendar Application
Frontend Services
Backend Services
User Interface
Authentication Service
Calendar Module
Event manager
Reminder Services
Database
Create Default Home Page
Create Login Page
Create default calendar page
Fetch credentials from login page
Credential validation
Populate deafult calendar
Fetch events from event manager and Reminder service
Update frontend UI
Establishing Database Connection
Querying and populating DB
Deliver Events to Calendar Module
Establishing Database Connection
Poll DB for upcoming reminders
Deliver Reminders to Calendar Module
Initialise Database
Initialise tables for events and reminders

Fig 2.1

2.4 Deliverables

The following are build components because they will be developed ground up:

• **Front end UI**: A clean frontend for the web application built using React (A JavaScript Framework). This will be what the user sees and directly interacts with. This component in turn communicates with the backend (Auth Service, Calendar Module, etc.) and delivers content in a clean and understandable way to the user.

• **Event manager**: This component is part of the backend (written in Flask) which is responsible to store per user event details. It interacts with a backend storage service (database) to hold these details. It additionally interacts with the calendar module to display upcoming events and the reminder service to give out reminder notifications.

• **Reminder Service**: This component is also part of the Flask backend and periodically polls the event manager to get details of upcoming events and notify users with these details accordingly.

The following are reuse components because they will be inherited either as 3rd party modules/libraries or as services entirely:

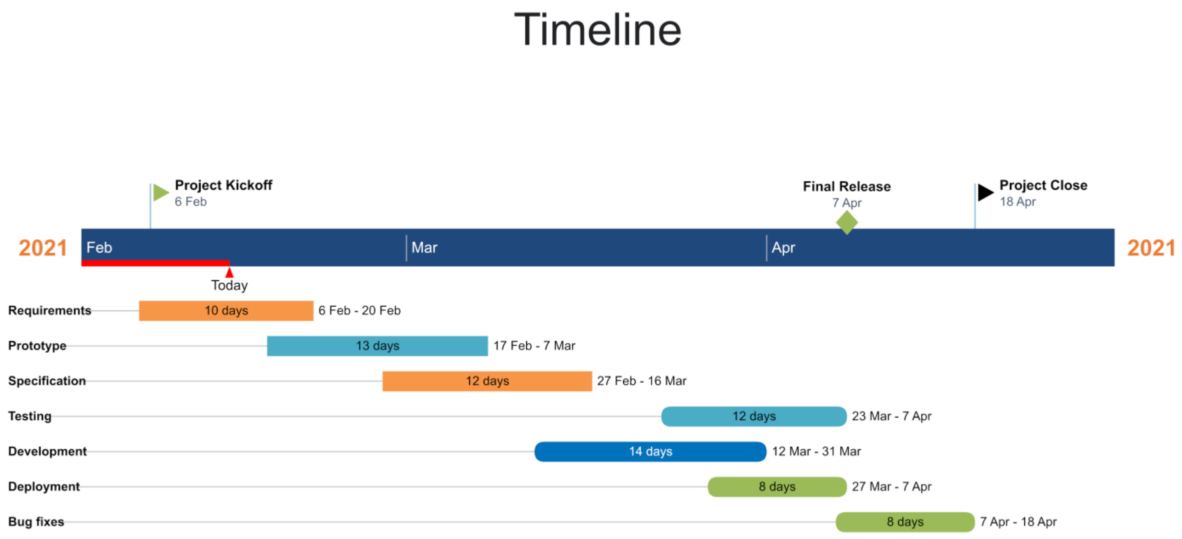
• **Authentication Service**: This service runs to authenticate user credentials provided through the front end. Either a 3rd party authentication application will be used or a simple service will be built which validates credentials with a backend database. This component is vital to ensuring user data segregation and confidentiality of information, thus will have to be developed with secure paradigms and handle sanitized input.

- **Calendar Module**: This component is the core service of the application; it provides calendar UI components along with the core structure of the calendar and all related aspects. It will be implemented using React's FullCalendar module. This module will interact with other services to deliver user calendar and events to the frontend.

2.5 Approximate Effort Required

- Requirements: 10 days/4 person-hours
- Prototype: 13 days/6 person-hours
- Specification: 12 days/5 person-hours
- Development: 14 days/8 person-hours
- Testing: 12 days/5 person-hours
- Deployment: 8 days/4 person-hours
- Bug Fixes: 8 days/6 person-hours

2.6 Timeline



3. Design Diagrams

3.1 High-Level Design Diagram(HLD)

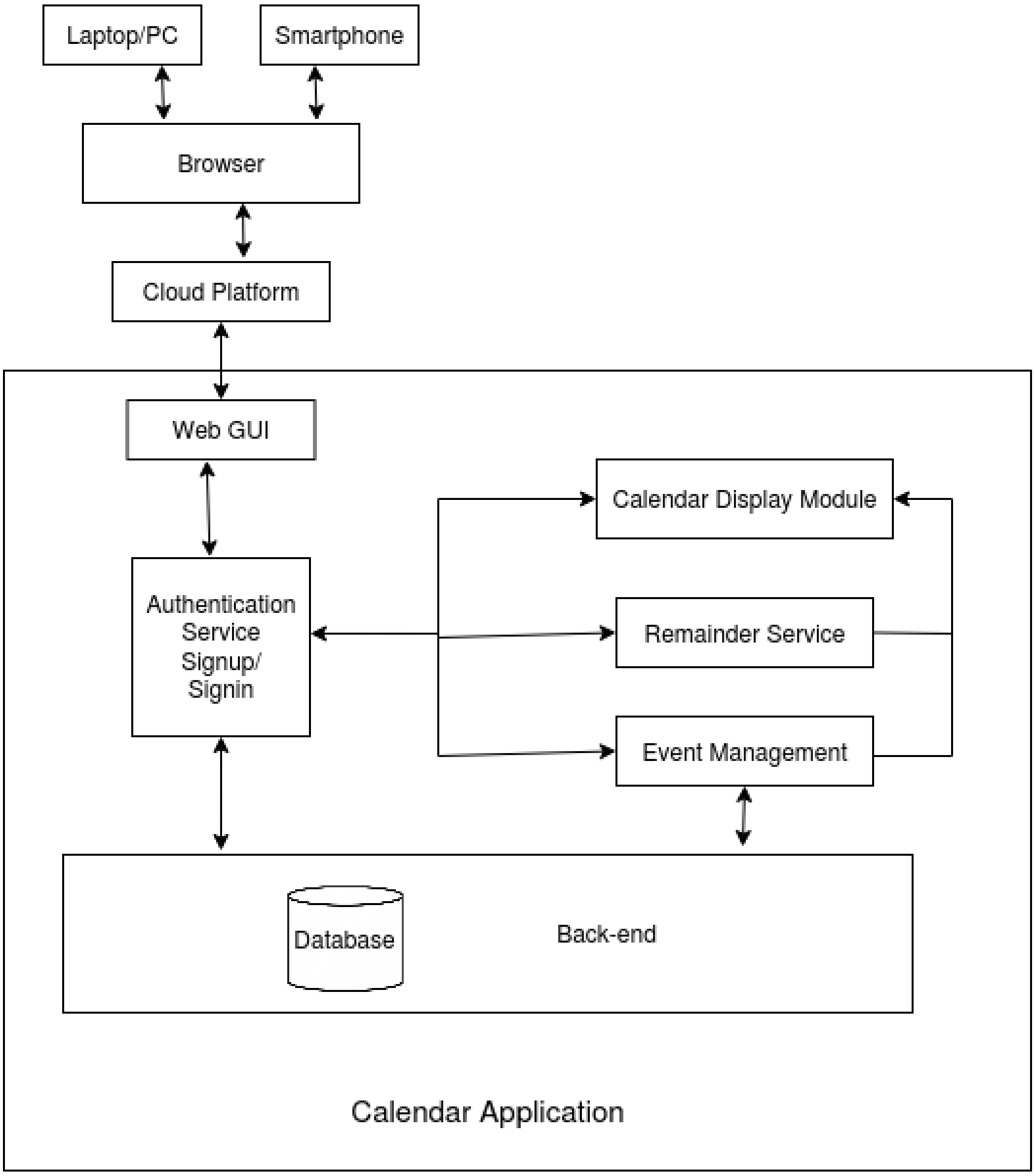


Fig. 3.1 High Level Design Diagram

The above diagram describes the high level characteristics of our project. The user can access the web application through laptop or any smartphone using a compatible browser. The Web UI first performs basic authentication such as signup/log in. On successful login the user can view their calendar through the display(view) module, set/delete events and include reminders for selected events. The backend consists of a database for storing user details for authentication and their respective event list.

3.2 Class diagram

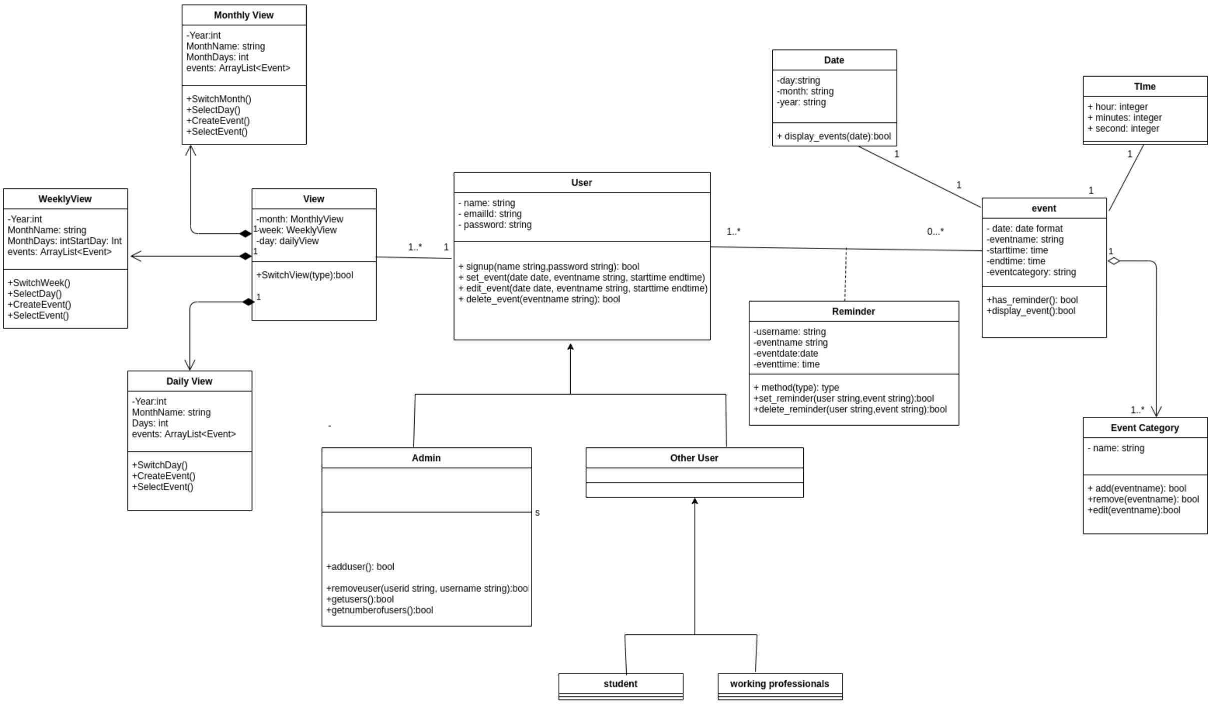


Fig. 3.2 Class Diagram

The class diagram above shows the various classes that are identified for the Calendar Application and the relationships they share with each other. The User class is responsible for storing information about each user, and handles the responsibility of registering the user, setting an event, and deleting the event. The Event class is responsible for storing the event details set by the user and supports has_reminder and display_event methods. The View class is responsible for displaying the various views of the Calendar Application - Monthly, Weekly and Daily View classes are specialized from the View class.

3.3 Activity Diagram

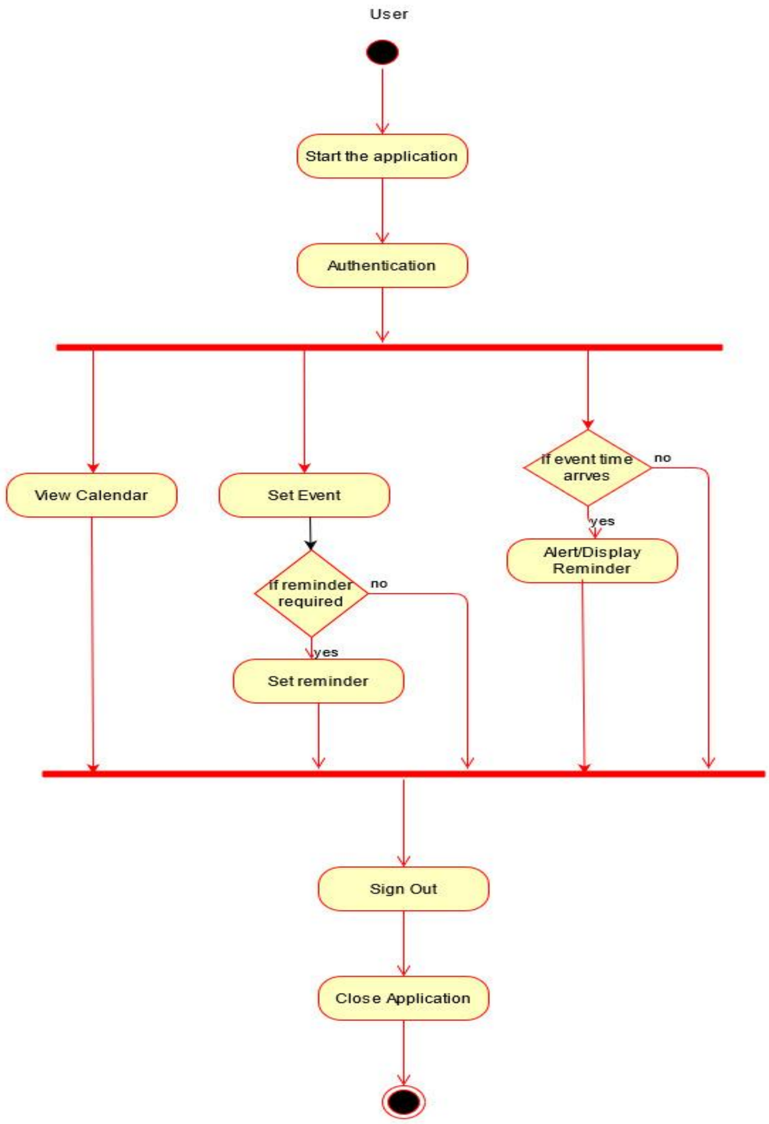


Fig. 3.3 Activity Diagram for the entire Application

The user first has to start the application login to his/her account , following which he can do any of the following:

- View calendar: The user can view the calendar day-wise, week-wise, and on a monthly-basis.
- Set event: The user can set an event on any day and also include an optional reminder.
- Once the reminder time is approximately equal to the start time of the event the user gets an email notification.
- Finally after using the application the user signs out of the application and closes it which marks the end of his activity.

3.4 Sequence Diagram

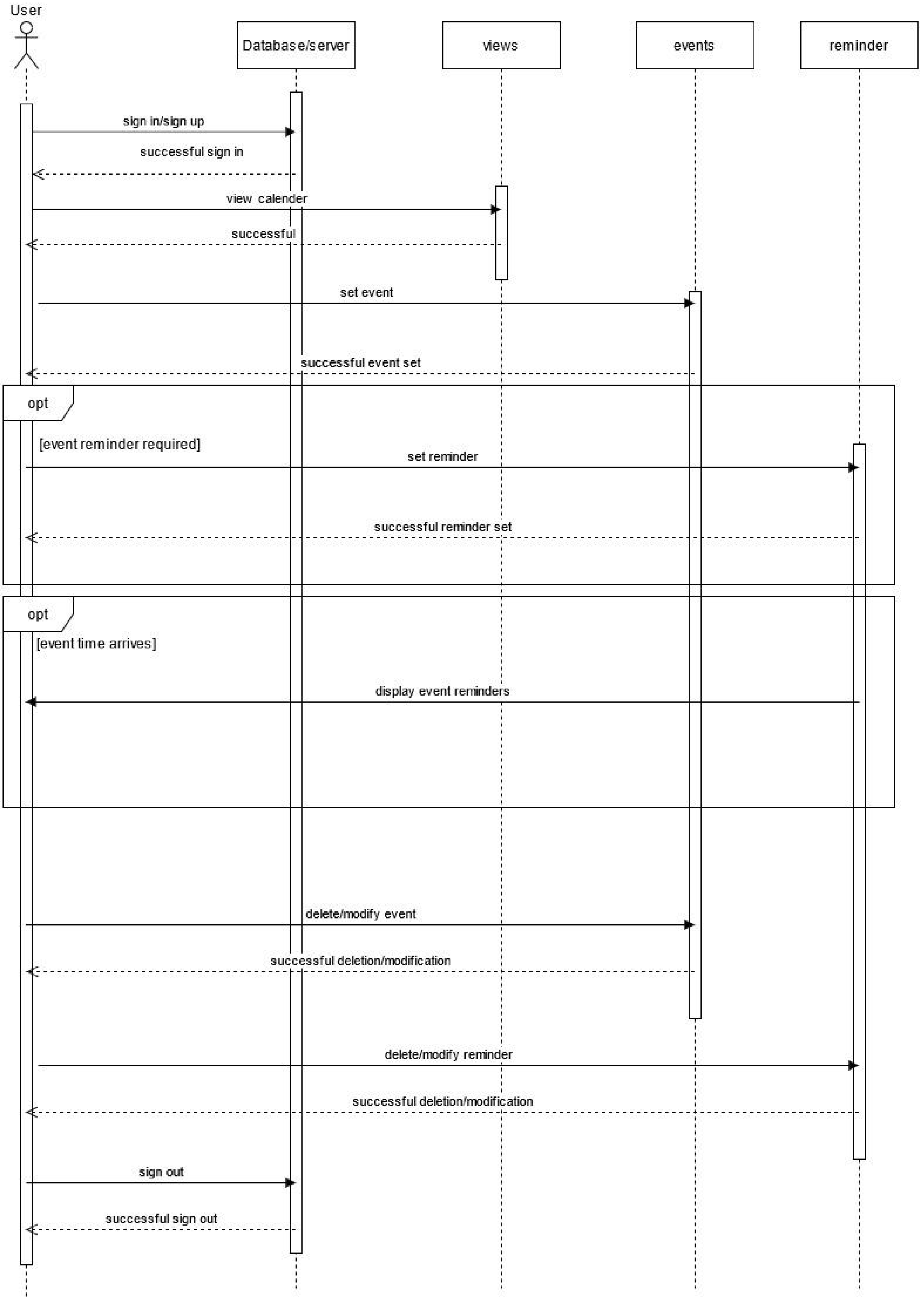


Fig. 3.4 Sequence Diagram for the entire application

The sequence diagram clearly indicates the order in which the activities are performed throughout a timeline. The user first logs in; on successful authentication the user receives a reply from the database. Then the user can perform any of the 3 activities concurrently:

- Views: gets a successful reply from the views module (day/week/month) on proper viewing.
- Set event: Once the event has been set, the user gets a reply from the events module confirming the same.
- Reminder : He can include an optional reminder for each event, in such cases the reminder module will send a definite reply.

Finally the user can sign out and on successful sign out , the database sends a reply to the user.

3.5 State Diagram

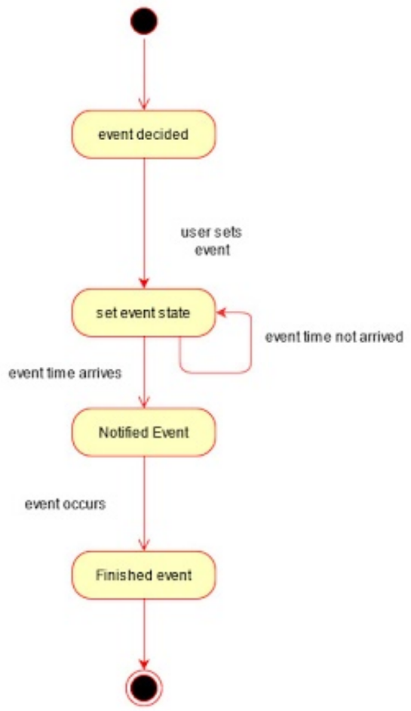


Fig. 3.5 State Diagram for the Events Use Case

The state diagram shows the states in which the event object goes through during the application. Once the event is added, the event object will be in a decided state. Later when the event time arrives, the event object reaches the notified state and finally reaches the finished state once the event occurs.

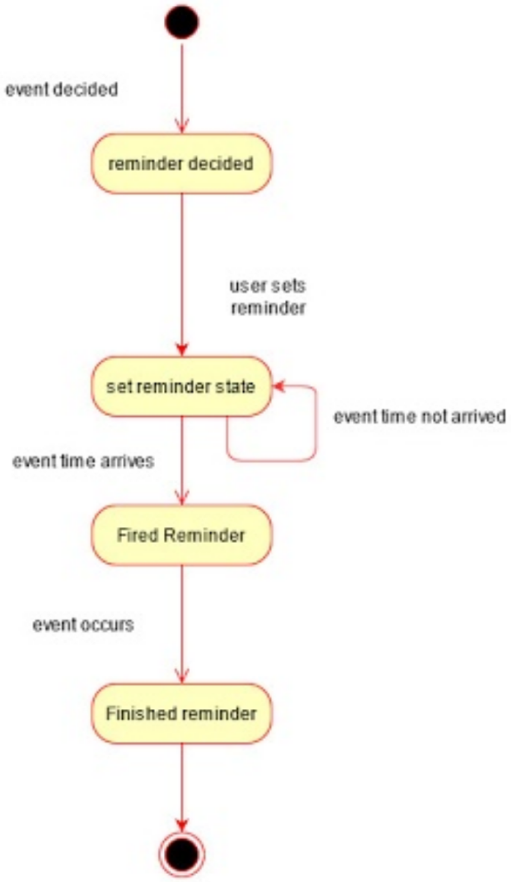


Fig. 3.6 State Diagram for the Reminders Use Case

The state diagram shows the states in which the reminder object goes through during the application. Once the reminder is set by the user, the reminder object will be in a reminder set state. It will continue to remain in the reminder set state as long as the event time has not arrived. Once it arrives, the state goes to reminder fire. And finally, it reaches the finished reminder state once the event occurs.

3.6 Module-wise sequence and activity diagrams :

3.6.1 Views

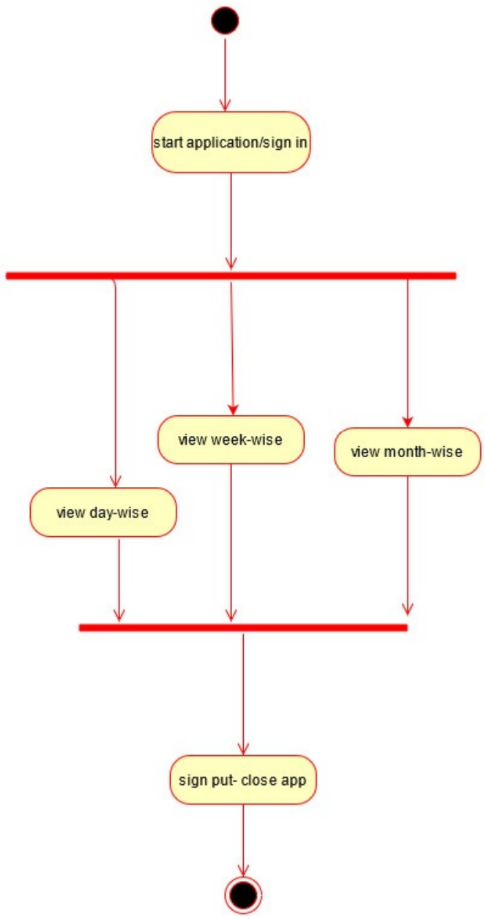


Fig. 3.7 State Diagram for the Views Use Case

The user has to start the application and perform basic authentication i.e login inorder to use the application. On successful login, the user is allowed to view the calendar day-wise, week-wise or on a monthly basis through the display(view) module. Since all the three can be performed in any order they are indicated by a fork and finally on completion of the viewing the calendar, the user signs out and closes the application.

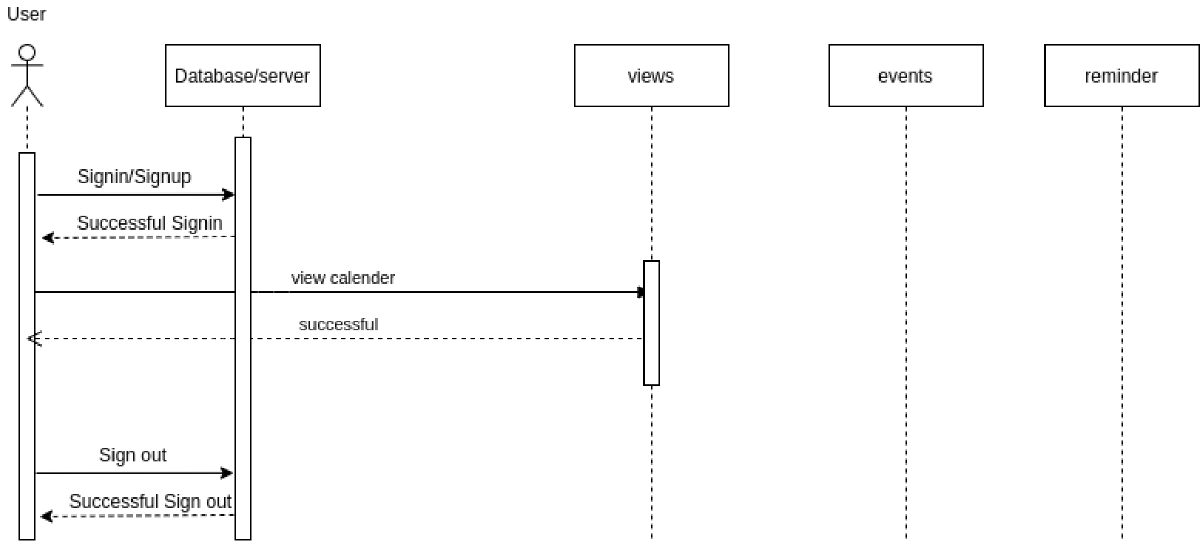


Fig. 3.8 Sequence Diagram for the Views Use Case

The sequence diagram clearly indicates the order in which the activities are performed throughout a timeline. The user first logs in; on successful authentication the user receives a reply from the database. Then the user can view the calendar on a daily, weekly or a monthly basis. He then receives a successful reply from the views module (day/week/month) on proper viewing.

3.6.2 Set Event

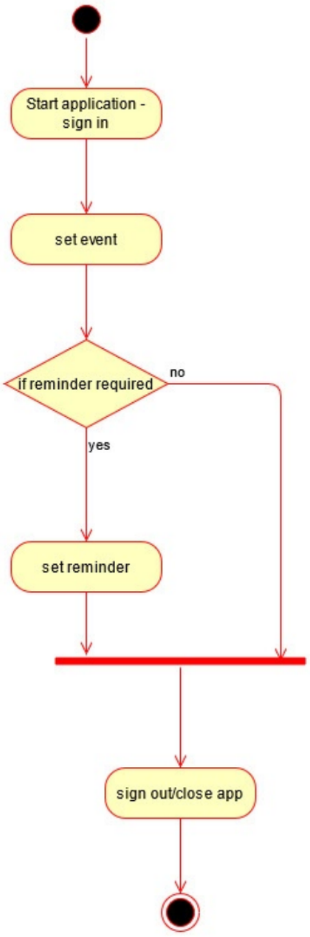


Fig. 3.9 State Diagram for the Events Use Case

The user starts the application and signs into his/her account, following which the user clicks the “Set Event” button which opens up a form for the user to fill. Upon filling the details of the event and setting an optional reminder for the event, the user submits the form and the event gets added to the user’s calendar. The user may continue to perform other activities on the application or may choose to sign out at this point.

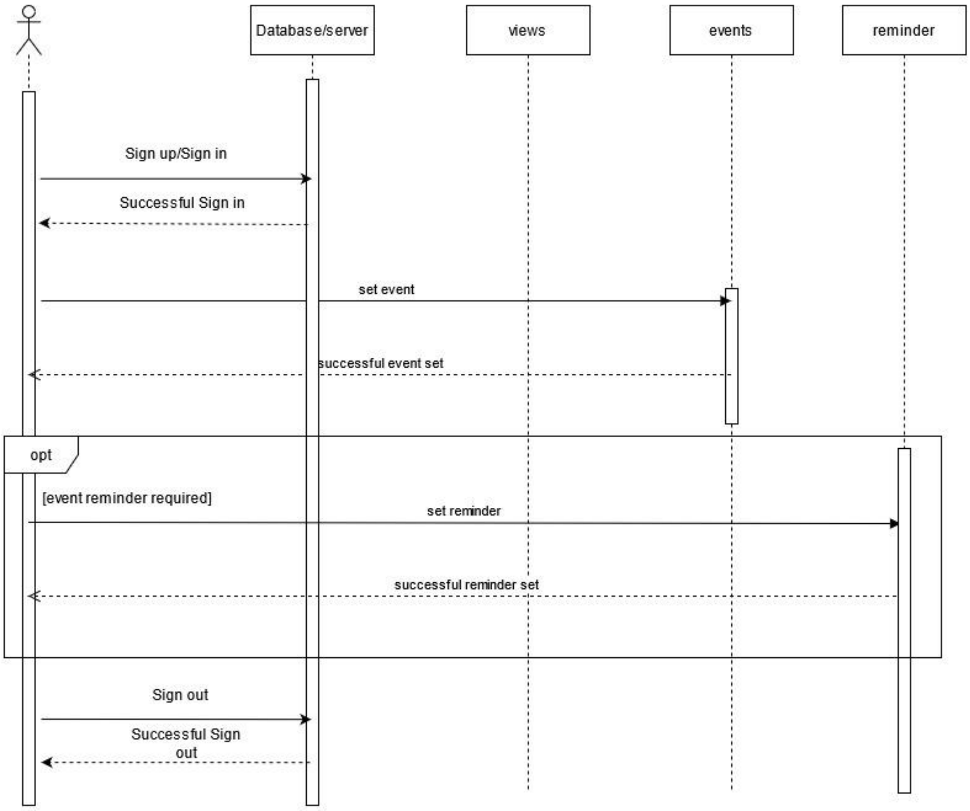


Fig. 3.10 Sequence Diagram for the Events Use Case

The user starts the application and signs into his/her account, following which the user clicks the “Set Event” button which opens up a form for the user to fill. Upon filling the details of the event and setting an optional reminder for the event, the user submits the form and the event gets added to the user’s calendar. The user may continue to perform other activities on the application or may choose to sign out at this point.

3.6.3 Get email Notification

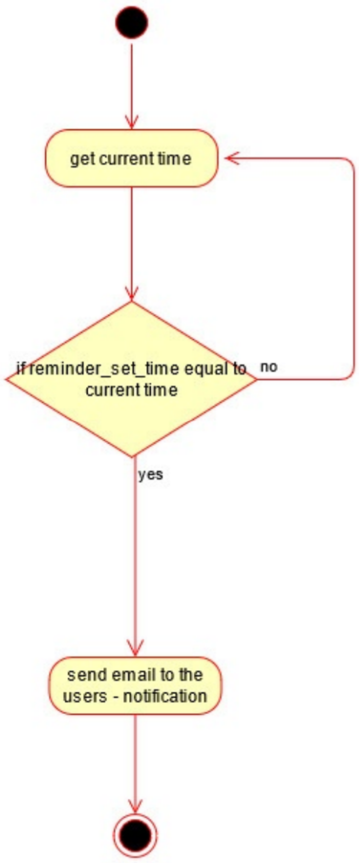

get current time
ifreminder_set_time equal to current time
no
yes
send email to the users - notification

Fig. 3.11 State Diagram for the Reminders Use Case

- Once the reminder time is approximately equal to the start time of the event the user gets an email notification to attend the event.

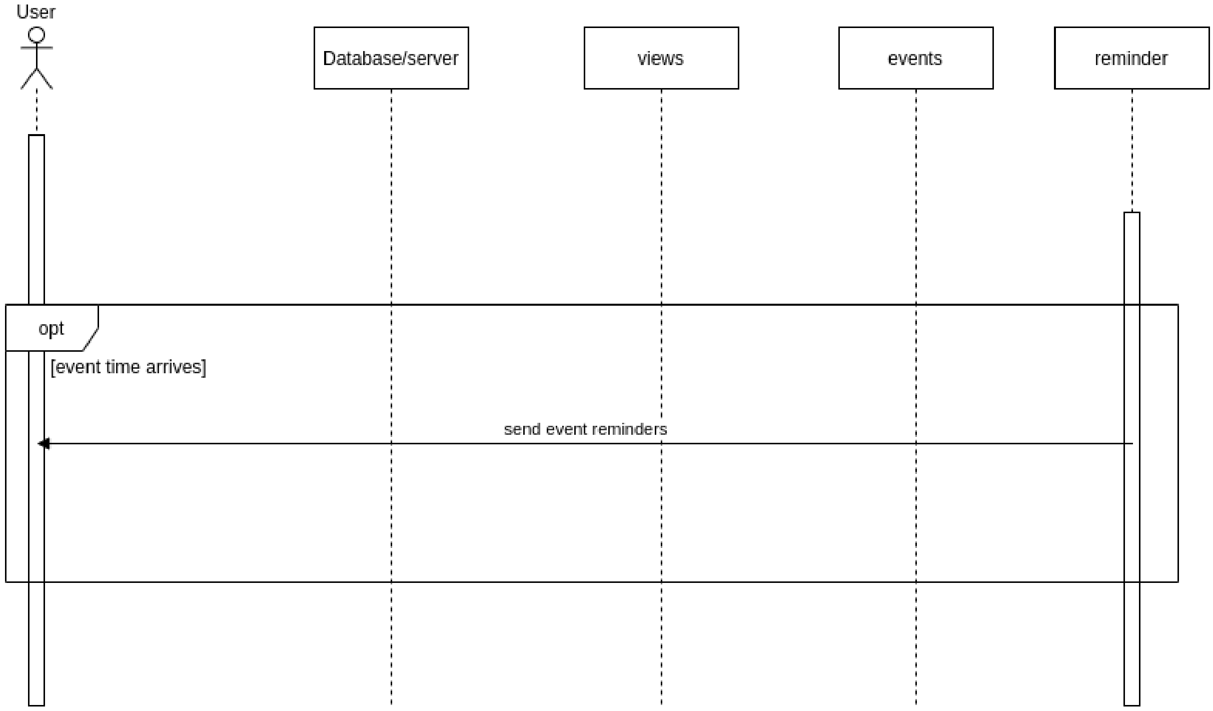


Fig. 3.12 Sequence Diagram for the Reminders Use Case

- Once the reminder time is approximately equal to the start time of the event the user gets an email notification to attend the event.

4. MODULE DESCRIPTION

4.1 Calendar Views

The calendar views module has been implemented where each user can have a daily, weekly and monthly view. Each view has been implemented in the form of a class with corresponding data members and member functions. The data members include basic information such as today's date, day, name of the month and the member function get_events() is used to get all the events of the particular user that come under that particular day for the day() module, particular week for the week() module and that respective month for the month() module.

For a particular user , who has signed up to use or application, on logging in to the application, he will be able to see the views section and can choose among the daily, weekly and monthly, the result of which is a list of events scheduled on that particular day, week and month respectively.

4.2 Setting/Deleting Events

Setting an event module has been implemented in the form of a class which allows the user to add an upcoming event. The user has to fill in the details of the event title, start date of the event, end date of the event and must also decide whether he wants to set an optional reminder, which will notify the user of the event on his/her registered email account 1 day before the event, all of which serve as the data members of the class. Once he submits the details, the said event will be recorded by the application and will be stored in the respective user's .json file which is a part of the add_event member function. The .json file is unique to every user and acts as a backend database for the application. To conclude, this module helps users to keep track of all the upcoming events in an easy manner.

Similarly we have a delete_event() member function where the user specifies the name of the event to be deleted and the respective event gets removed from the .json file of that particular user.

4.3 Sending Email Reminders

The sending email reminders module has been implemented which will send reminders to users on their registered email-id. The user while adding an event has to decide whether he wants to set an optional reminder for the event. If the user wishes to set the reminder field, then the user will receive a reminder on his/her registered email account 1 day before the event is scheduled to begin.

For implementation purposes, we have created a new thread whenever a user logs in to the application. The thread will start and will run an infinite loop going over all the events in the respective user's .json file, which will have the events added by the user. Emails will be sent to the user's registered account 1 day before the scheduled event is set to begin.

5. Test Cases

5.1 Views

Test Case ID	Name of Module	Test case description	Pre-conditions	Test Steps	Test data	Expected Results	Actual Result	Test Result
UT-01	Daily View	To test the daily view functionality	Access to Chrome Browser, login and click on Daily View under Views section	1: Start the Calendar Application 2: Login by entering Email and Password. Click on views 3: Select daily view	Email Id : v@gmail.com Password : visheeeee	Daily view of the user's calendar displayed on the screen	Daily view of the user's calendar displayed on the screen	Pass
UT-02	Weekly View	To test the Weekly view functionality	Access to Chrome Browser, login and click on Weekly View under Views section	1: Start the Calendar Application 2: Login by entering Email and Password. Click on views 3: Select weekly view	Email Id : v@gmail.com Password : visheeeee	Weekly view of the user's calendar displayed on the screen	Weekly view of the user's calendar displayed on the screen	Pass
UT-03	Monthly View	To test the Monthly view functionality	Access to Chrome Browser, login and click on Monthly View under Views section	1: Start the Calendar Application 2: Login by entering Email and Password. Click on views 3: Select monthly view	Email Id : v@gmail.com Password : visheeeee	Monthly view of the user's calendar displayed on the screen	Monthly view of the user's calendar displayed on the screen	Pass
UT-04	Yearly view	To test the Yearly view functionality	Access to Chrome Browser,	1: Start the Calendar Application	Email Id :	Yearly View of the	Yearly View of the	Pass

			login and click on yearly View in Views	2: Login by entering Email and Password. Click on views 3: Select Yearly View if option is available	v@gmail.com Password : visheee	user’s calendar not displayed since that functionality does not exist.	user’s calendar not displayed since that functionality does not exist.	
IT-05	Viewing events in the daily view of the calendar	To check if the events set by the user are visible in the user’s daily view of the calendar	Access to Chrome Browser, login and click on daily View under Views section	1: Start the Calendar Application 2: Login by entering Email and Password. Click on views 3: Click on daily view	Email Id : v@gmail.com Password : visheee	User is able to view the events of the current day	User is able to view the events of the current day	Pass
IT-06	Viewing events in the weekly view of the calendar	To check if the events set by the user are visible in the user’s weekly view of the calendar	Access to Chrome Browser, login and click on weekly View under Views section	1: Start the Calendar Application 2: Login by entering Email and Password. Click on views 3: Click on weekly view	Email Id : v@gmail.com Password : visheee	User is able to view the events of the current week	User is able to view the events of the current week	Pass
IT-07	Viewing events in the monthly view of the calendar	To check if the events set by the user are visible in the user’s monthly view of the calendar	Access to Chrome Browser, login and click on monthly View under Views section	1: Start the Calendar Application 2: Login by entering Email and Password. Click on views 3: Click on monthly view	Email Id : v@gmail.com Password : visheee	User is able to view the events of the current month	User is able to view the events of the current week	Pass

IT-08	Viewing reminder details	To view the reminder configuration of any event	Access to Chrome Browser, login and click on any View under the Views section	1: Start the Calendar Application 2: Login by entering Email and Password. Click on views 3: Click on any view	Email Id : v@gmail.com Student. Password : visheee	User is only able to view the name of the event and the start date of any event and can't see if a reminder for that particular event has been set.	User is only able to view the name of the event and the start date of any event and can't see if a reminder for that particular event has been set.	Pass
UT-09	Navigating calendar	To check the navigation functionality	Access to Chrome Browser, login and click on any View and click on arrows to navigate if available	1: Start the Calendar Application 2: Login by entering Email and Password. Click on views 3: Click on any view 4. Click on left/right arrows	Email Id : v@gmail.com Password : visheee	User is unable to navigate through the calendar, only able to view the current week's, month 's or day's calendar.	User is unable to navigate through the calendar, only able to view the current week's, month 's or day's calendar.	Pass
ST-10	System testing from the	To check if every other use case	Access to Chrome Browser and login	1: Start the Calendar Application	**Log in** Email	User is able to see the	User is able to see the reflecte	Pass

	perspective of theViews Test Case.	works with this use case		2: Login by entering Email and Password. 3. Click on Add Event, enter the event details and click submit. Go back to the home page and check any view,the event name will be displayed wherever applicable. 4. Click on Delete Event after logging in, enter the name of the event and click submit. The event will not be displayed anymore under any view. 5. Look out for Email Reminders for upcoming events when logged in.	Id : v@gmail.com Student. Password : visheee **Event Details** Event name: birthday Startdate: 2021-04-17 End Date: 2021-04-18 Require reminder 1)yes 2)no **Delete Event:**	reflected changes wherever applicable.	d changes wherever applicable.	

					Event Name: birthday			

5.2 Events

Test Case ID	Name of Module	Test case description	Pre-conditions	Test Steps	Test data	Expected Results	Actual Result	Test Result
UT-11	Creating an event	To check if the user is able to create an event and mark it on their calendar	Access to Chrome Browser, login and click on Add Event	1: Start the application 2: Login by entering Email and Password. 3: Click on Add Event. 4. Fill the form presented to the user and submit.	**Login** :Email Id : v@gmail.com Student. Password : visheee **Event Details** Event name: birthday Startdate: 2021-04-17 EndDate: 2021-04-18 Require reminder 1)yes 2)no	Event should be created and marked on the daily, weekly, monthly view of the user's calendar.	Event is created and marked on the daily, weekly and monthly view of the user's calendar.	Pass
UT-12	Editing Event	To check the edit functionality	Access to Chrome Browser,	1: Start the Calendar	**Login** Email Id :	Edit should allow	Edit functiona	Fail

			login and click on Edit Event	Application 2: Login by entering Email and Password. 3: Click on Edit Event. 5. Fill the form presented to the user and submit.	v@gmail.com Student. Password : visheee	the user to edit event details like date, start time and end time	lity does not exist	
UT-13	Valid start and end time	To check if the start time and end time are valid	Access to Chrome Browser, login and click on Add Event	1: Start the Calendar Application 2: Login by entering Email and Password. 3: Click on add Event. 5. Fill the form presented to the user and submit.	**Login** :Email Id : v@gmail.com Student. Password : visheee **Event Details** Event name: birthday Startdate: 2021-04-17 EndDate: 2021-04-18 Require reminder 1)yes 2)no	Error if end time is less than start time	No error if end time is less than start time	Fail

UT-14	Deleting event	To check the delete functionality	Access to Chrome Browser, login and click on Edit Event	1: Start the Calendar Application 2: Login by entering Email and Password. 3: Click on Delete Event. 4.Fill the form presented to the user and submit	**Login** Email Id : v@gmail.com Student. Password : visheee **Delete Event :** Event Name: birthday	Delete should allow the user to delete the event mentioned in the form.	User is able to delete the event using event name	Pass
UT-15	Sharing the event among users	To check if multiple users can share the same event details.	Access to Chrome Browser, login and click on Add Event.	1: Start the Calendar Application 2: Login by entering Email and Password. 3: Click on Add Event. 4: Add users to the event if the option is available in the form presented to the user creating the event.	Email Id : v@gmail.com Student. Password : visheee	The users involved in the event can view the daily, weely, monthly views of their respective calendars.	Since the user creating the event can't add other users to the event, the event will only show up on the user's monthly view of his/her calendar and no one else's.	Fail.
UT-16	Duplication of events	To check if the user is able to	Access to Chrome Browser,	1: Start the Calendar	**Login** :Email Id :	The duplicate event	Since the functionality	Fail

		duplicate any event created before.	login and click on add event and add the same event previously added.	Application 2: Login by entering Email and Password 3: Click on Add Event. 4. Fill the form presented to the user and submit.	v@gmail.com Student. Password : visheee **Event Details** Event name: birthday Startdate: 2021-04-17 EndDate: 2021-04-18 Require reminder 1)yes 2)no	should not be added to the user’s events	doesn’t exist, the testcase fails and the same event is added once again.	
UT-17	Add location details	To check if the location can be set for any event taking place.	Access to Chrome Browser, login and click on Add Event.	1: Start the Calendar Application 2: Login by entering Email and Password. 3: Click on Add Event. 4: Look for the Add location option in	Email Id : v@gmail.com Student. Password : visheee	The location will be set for every event taking place on the user’s calendar.	Since the functionality doesn’t exist, none of the events have any location data associated with it.	Fail.

				the form presented to the user.				
UT-18	Set the event to repeat as required.	To set the event to repeat as and when required.	Access to Chrome Browser, login and click on Add Event.	1: Start the Calendar Application 2: Login by entering Email and Password. 3: Click on Add Event. 4: Look for the option to set the event to repeat.	Email Id : v@gmail.com Student. Password : visheee	The event will be marked as and when specified to repeat.	Since the functionality doesn't exist, no event will ever repeat unless explicitly set by the user.	Fail.
IT-19	Set a reminder to each event	To check if the remainder setting functionality	Access to Chrome Browser, login and click on Add Event.	1: Start the Calendar Application 2: Login by entering Email and Password 3: Click on Add Event. 4. Fill the form presented to the user and click on yes in reminder required and submit	**Login** :Email Id : v@gmail.com Student. Password : visheee **Event Details** Event name: birthday Startdate: 2021-04-17 EndDate: 2021-04-18	The event should have the reminder as set	The reminder will be set for the event as indicated in the json file	Pass

					Require reminder 1)yes 2)no			
IT-20	Not setting a reminder to an event	To check if the remainder setting functionality	Access to Chrome Browser, login and click on Add Event.	1: Start the Calendar Application 2: Login by entering Email and Password 3: Click on Add Event. 4. Fill the form presented to the user, click on no in reminder required and submit.	**Login** :Email Id : v@gmail.com Student. Password : visheee **Event Details** Event name: birthday Startdate: 2021-04-17 EndDate: 2021-04-18 Require reminder 1)yes 2)no	The event should not have the reminder as set	The reminder will not be set for the event as indicated in the json file	Pass

5.3 Reminders through Email

Test Case ID	Name of Module	Test case description	Pre-conditions	Test Steps	Test data	Expected Results	Actual Result	Test Result
UT-21	Sending Email reminders when the user is logged in	To check if the user is receiving Email reminders when the user is	Access to Chrome Browser, and logging into the user's account	1: Start the Calendar Application	Email Id : v@gmail.com Student.	Emails will be sent to the user's registered Email account if	Emails have been sent to the user's register	Pass

		logged in to his/her account.		2: Enter Email ID and Password 3: Click Submit	Password : visheee	the reminder for any particular event has been set.	edEmail account if the reminder for any particular event has been set.	
UT-22	Sending Email reminders when the user is not logged in	To check if the user is receiving Email reminders when the user is logged in to his/her account.	Access to Chrome Browser, and logging into the user's account	1: Start the Calendar Application 2: Enter Email ID and Password 3: Click Submit	-	Emails will not be sent to the user's email		Fail
UT-23	Setting reminder	To check if the user is able to set reminders for any of the user created events	Access to Chrome Browser, and logging into the user's account and selecting Add Reminder under the Add Event form.	1: Start the Calendar Application 2: Enter Email ID and Password 3: Click Submit 4: Click on Add Event 5: Click	**Login** :Email Id : v@gmail.com Student. Password : visheee **Event Details** Event name: birthday Startdate:	Success message displayed stating the event reminder has been set succesfully for the respective event.	Success message displayed stating the event reminder has been set succesfully for the respective event.	Pass

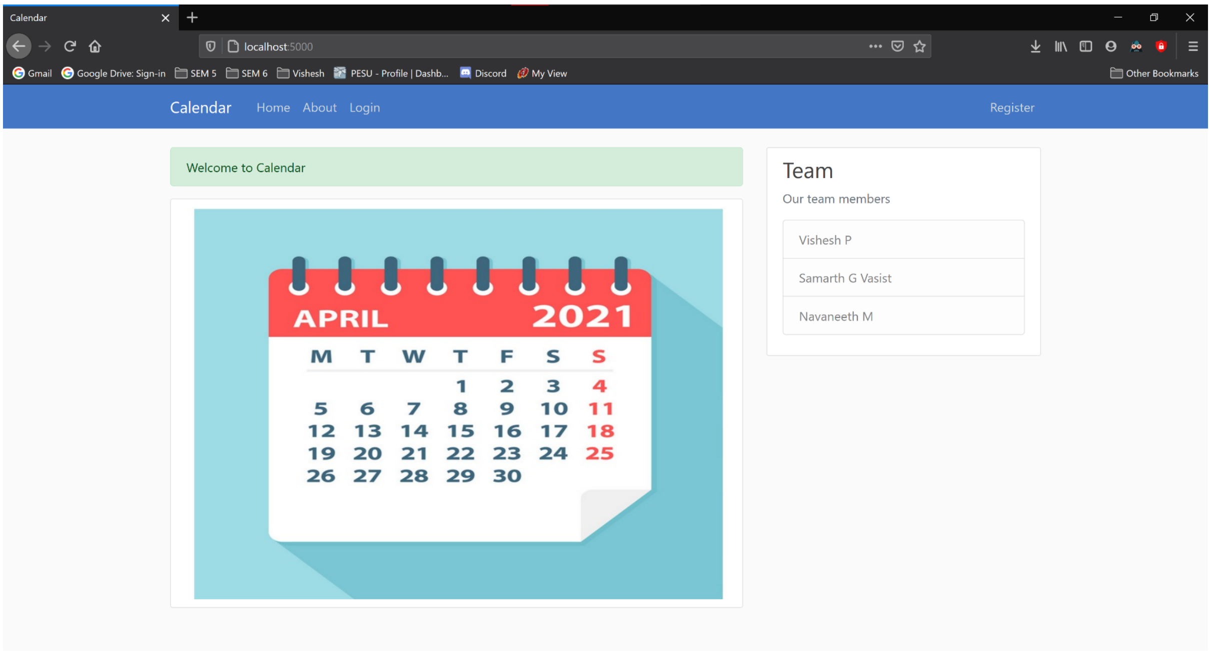
				on Add Reminder after filling the event details	2021-04-17 EndDate: 2021-04-18 Require reminder 1)yes 2)no			
UT-24	Deleting reminder	To check if the user is able to delete reminders set by him/her for any event created.	Access to Chrome Browser, and logging into the user’s account and selecting Delete Reminders.	1: Start the Calendar Application 2: Enter Email ID and Password 3: Click Submit 4: Click on Delete Reminders 5: Select the Event for which the user wishes to delete reminders.	Email Id : v@gmail.com Student. Password : visheee	Success message displayed to the user stating the reminder for the respective event has been deleted.	Reminder deletion functionality does not exist	Fail
UT-25	Sharing reminder	To check if the user is able to share the reminder among other users.	Access to Chrome Browser, and logging into the user’s	1: Start the Calendar	Email Id : v@gmail.com	The user can add other users to remind them	Since the functionality doesn’t exist,	Fail

			account and selecting Add Reminder under the Add Event form.	Application 2: Enter Email ID and Password 3: Click Submit 4: Click on Add Event 5: Click on Add Reminder after filling the event details 6: Look for sharing options	Student. Password : visheee	about the event.	the user cannot add other users to the reminder and hence no other user will be reminded.	
IT-26	Receiving reminder within bounded time when user is logged in	To check if the user is receiving reminders 1 day before the start time of the respective event.	Access to Chrome Browser, and logging into the user's account.	1: Start the Calendar Application 2: Enter Email ID and Password 3: Click Submit	Email Id : v@gmail.com Student. Password : visheee	The user gets reminders 1 day before the start time of the event.	The user gets reminders 1 day before the start time of the event.	Pass
IT-27	Receiving	To check if the user is	-	-	-	The user gets	The user	Fail

	reminder within bounded time when user is not logged in	receiving reminders 1 day before the start time of the respective event.				reminders 1 day before the start time of the event.	does not receive reminders 1 day before the start time of the event.	
IT-28	Sending message reminders to phone when the user is logged in	To check if the user receives reminders about the respective event on the user's phone number specified	Access to Chrome Browser, and log into the user's account.	1: Start the Calendar Application 2: Enter Email ID and Password 3: Click Submit 4: Click the Add Reminder option after filling in the event details. 5: Enter the phone number to which the user wishes to receive	Email Id : v@gmail.com Student. Password : visheee	The user will receive a reminder on his/her phone 1 day before the event starts.	The user does not receive a reminder on his/her phone 1 day before the event starts as this functionality does not exist	Fail

				reminders about the respective events.				
IT-29	Sending message reminders to phone when the user is not logged in	To check if the user receives reminders about the respective event on the user’s phone number specified		-	-	The user will receive a reminder on his/her phone 1 day before the event starts.	The user does not receive a reminder on his/her phone 1 day before the event starts.	Fail

6. Screenshots of Output



Welcome to Calendar

Team

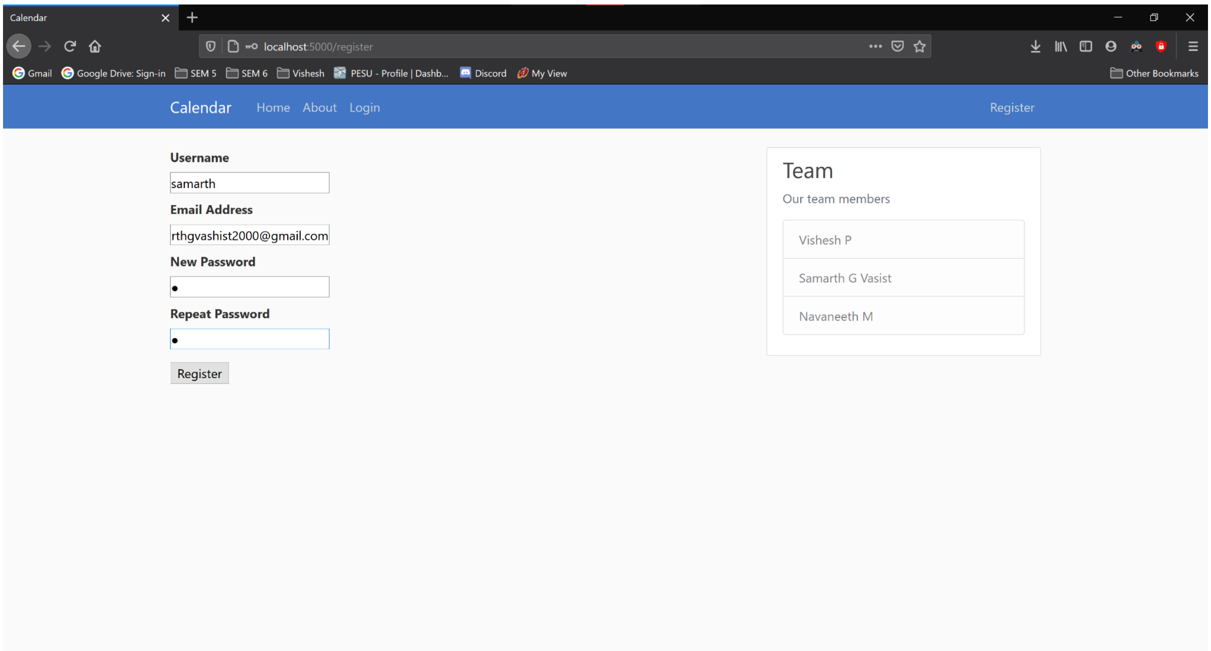
Our team members

Vishesh P

Samarth G Vasist

Navaneeth M

Fig. 6.1 Home Page



Username: samarth

Email Address: rthgvashist2000@gmail.com

New Password: •

Repeat Password: •

Register

Team

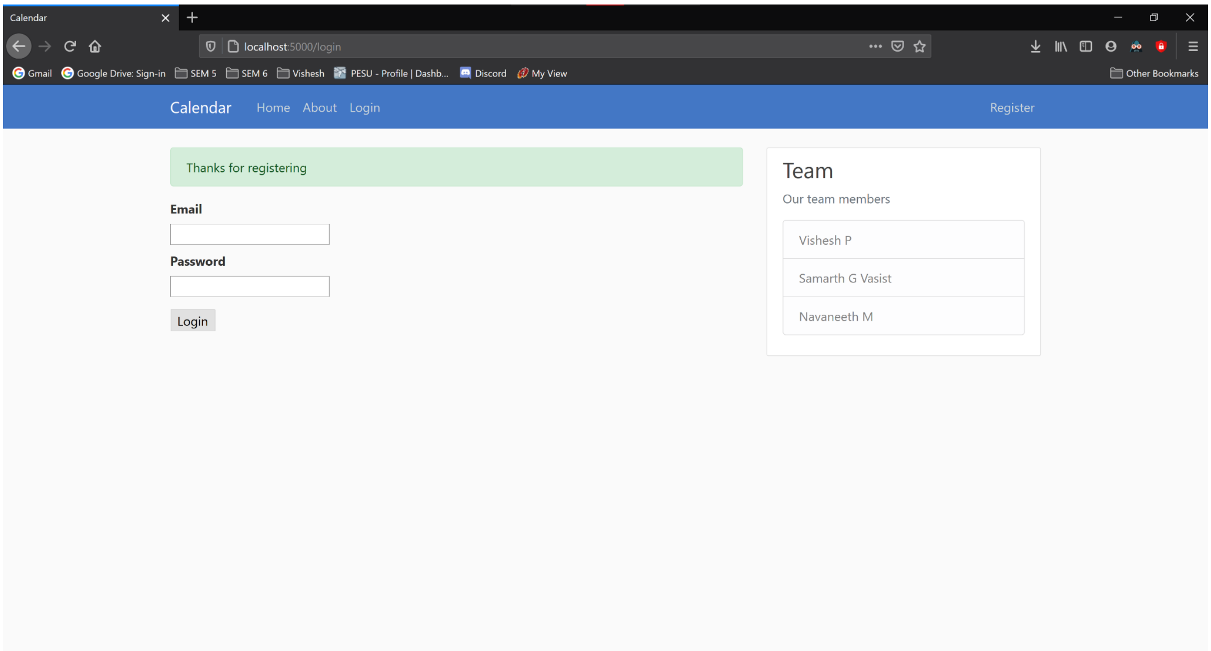
Our team members

Vishesh P

Samarth G Vasist

Navaneeth M

Fig. 6.2 Register Page



Thanks for registering

Email

Password

Login

Team

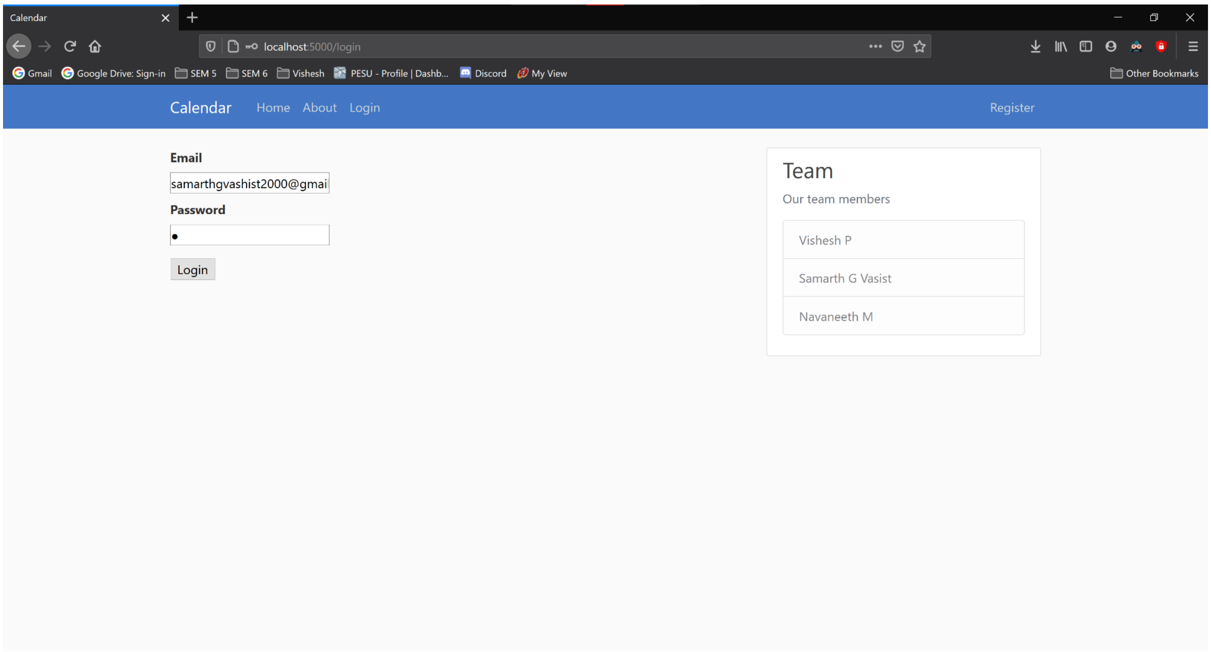
Our team members

Vishesh P

Samarth G Vasist

Navaneeth M

Fig. 6.3 Successfully Registered



Email

samarthgvashist2000@gmai

Password

•

Login

Team

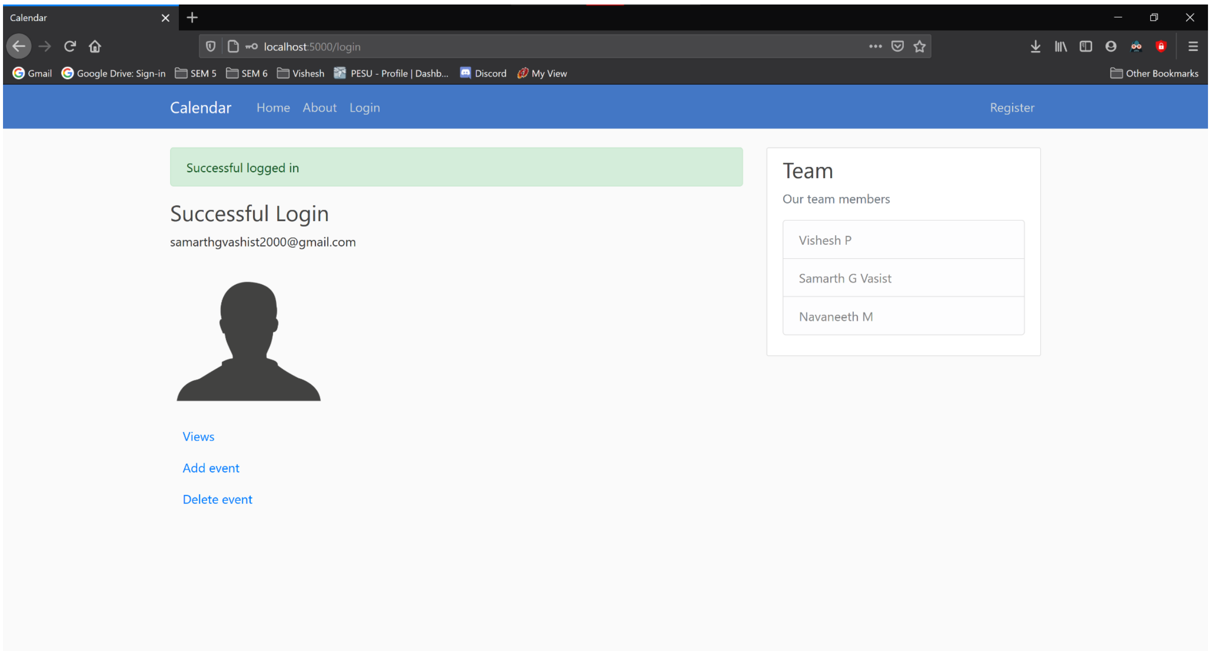
Our team members

Vishesh P

Samarth G Vasist

Navaneeth M

Fig. 6.4 Login Page



Successful logged in

Successful Login

samarthgvashist2000@gmail.com

Views

Add event

Delete event

Team

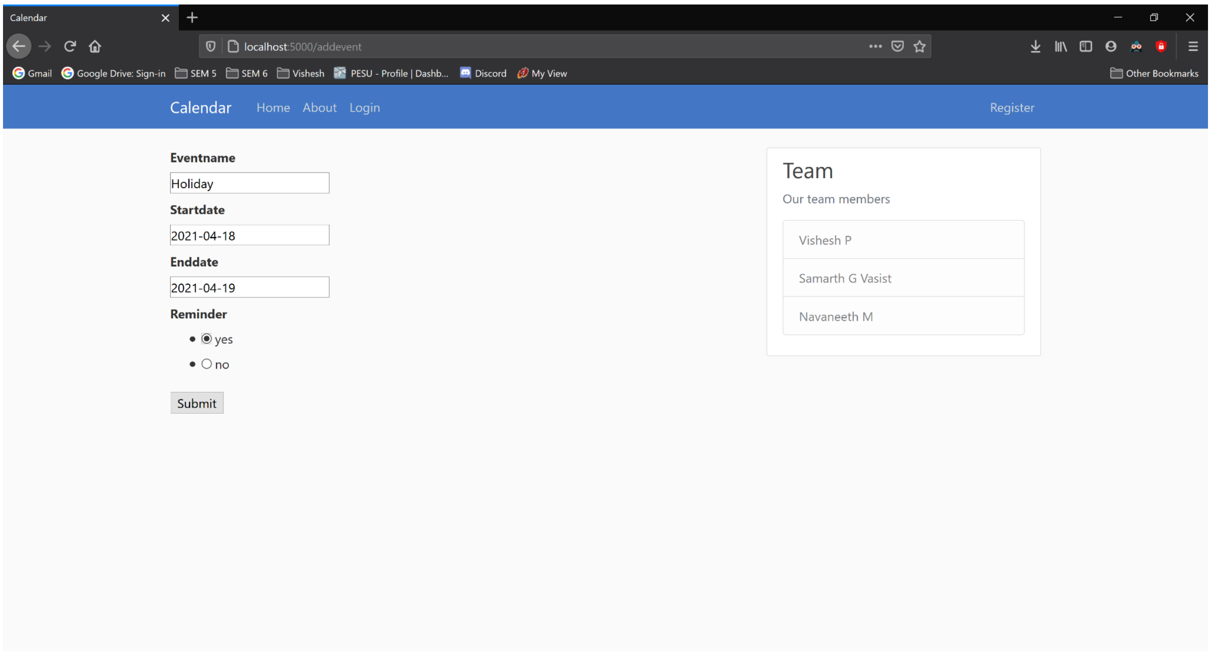
Our team members

Vishesh P

Samarth G Vasist

Navaneeth M

Fig. 6.5 Successfully logged in



Eventname

Holiday

Startdate

2021-04-18

Enddate

2021-04-19

Reminder

- ● yes
- ○ no

Submit

Team

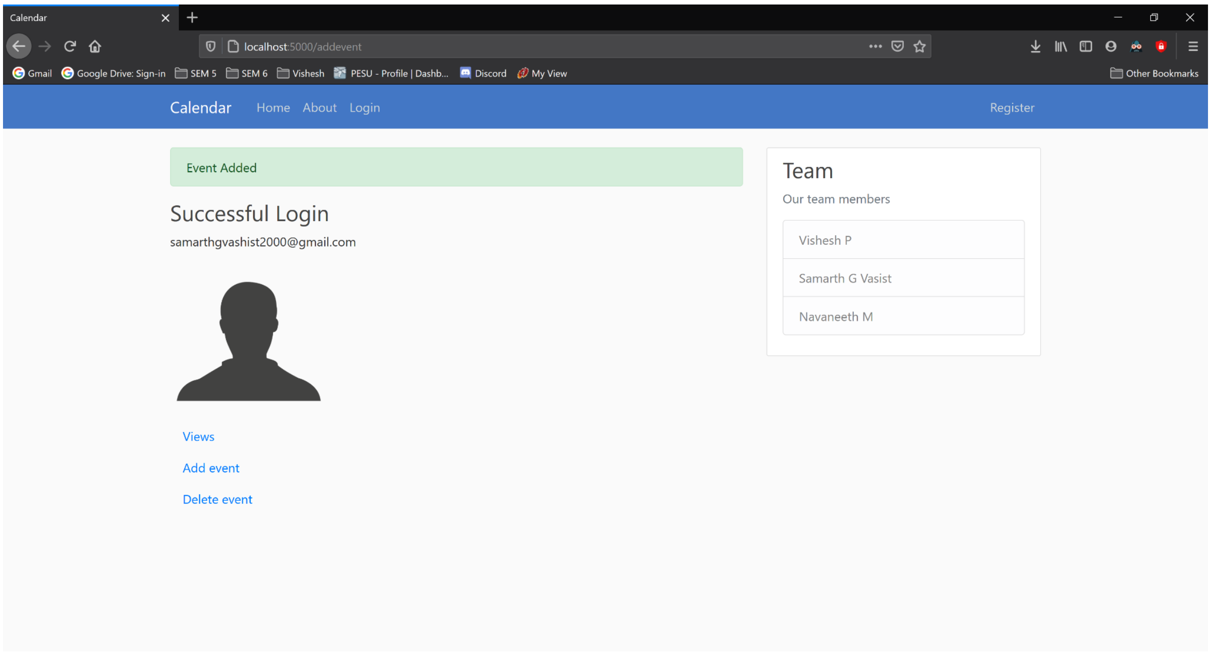
Our team members

Vishesh P

Samarth G Vasist

Navaneeth M

Fig. 6.6 Add Event



Event Added

Successful Login

samarthgvashist2000@gmail.com

Views

Add event

Delete event

Team

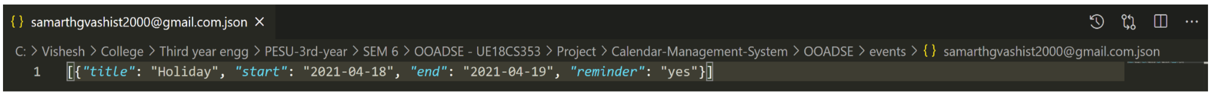
Our team members

Vishesh P

Samarth G Vasist

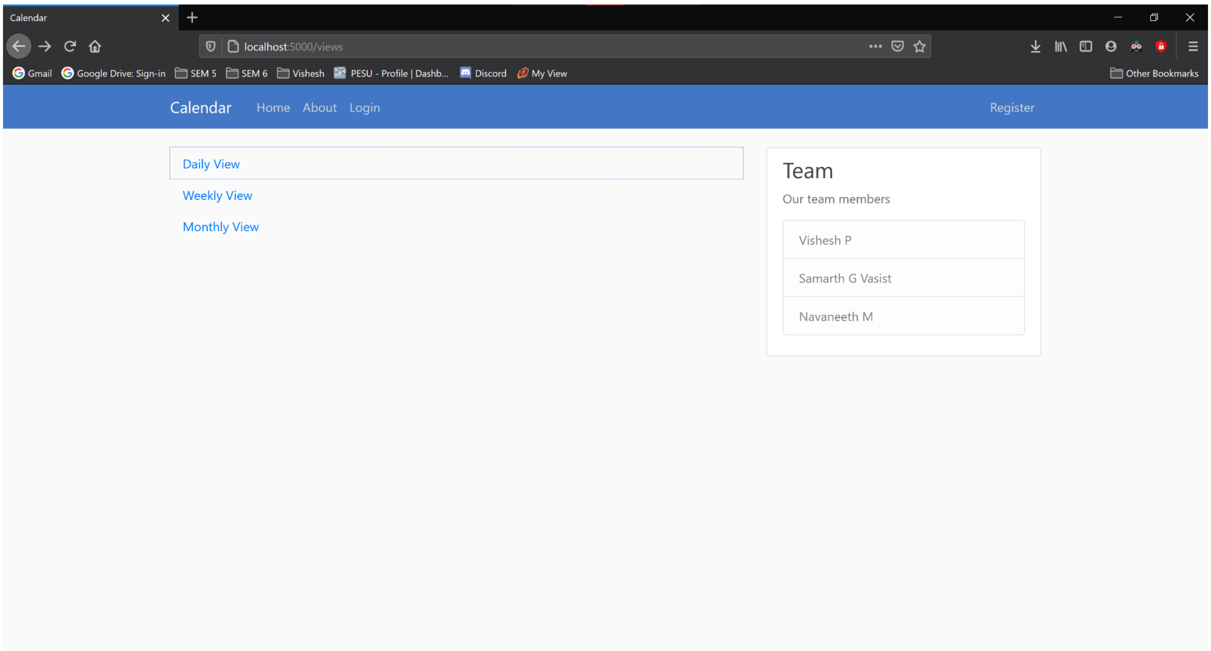
Navaneeth M

Fig. 6.7 Home Page upon successful event addition.


{ } samarthgvashist2000@gmail.com.json ×
C: > Vishesh > College > Third year engg > PESU-3rd-year > SEM 6 > OOADSE - UE18CS353 > Project > Calendar-Management-System > OOADSE > events > { } samarthgvashist2000@gmail.com.json

```
[{"title": "Holiday", "start": "2021-04-18", "end": "2021-04-19", "reminder": "yes"}]
```

Fig 6.8 samarthgvashist2000@gmail.com.json contents upon adding event



Daily View

Weekly View

Monthly View

Team

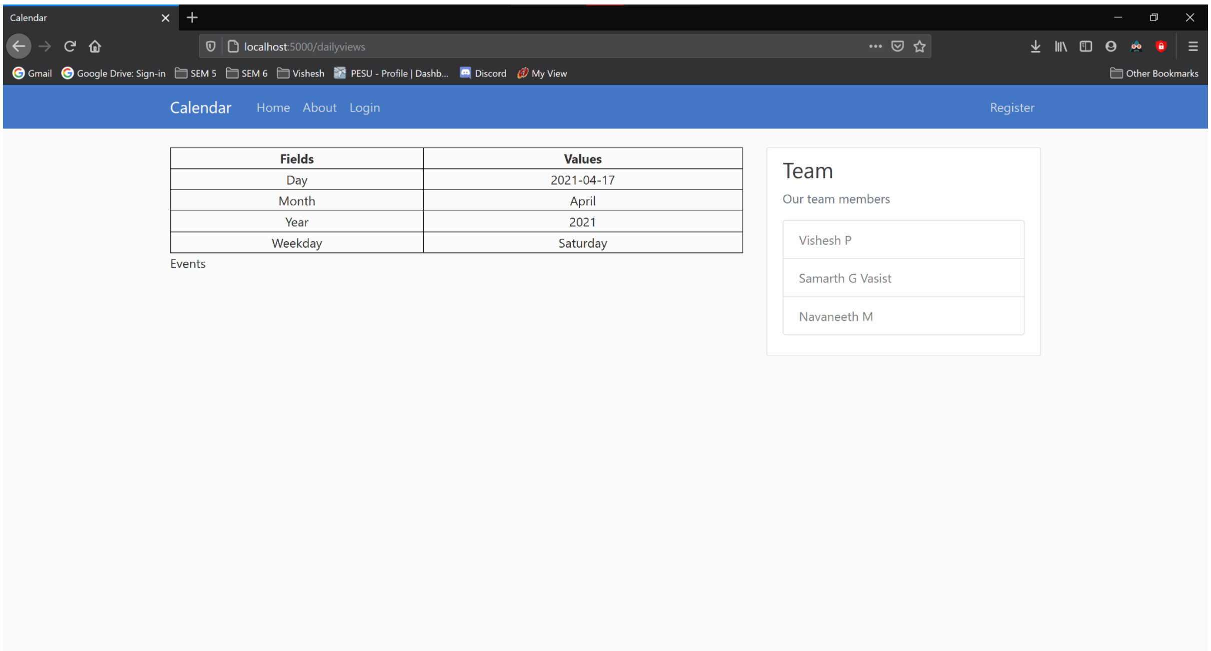
Our team members

Vishesh P

Samarth G Vasist

Navaneeth M

Fig 6.9 Views Page



Fields	Values
Day	2021-04-17
Month	April
Year	2021
Weekday	Saturday

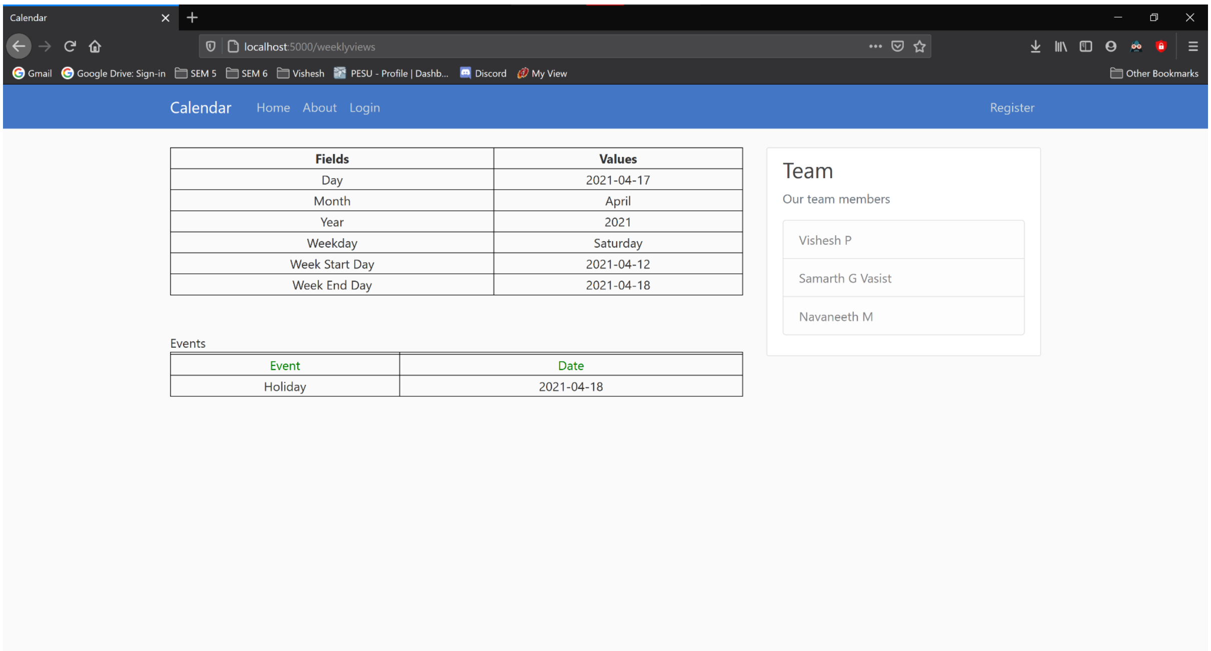
Events

Team

Our team members

- Vishesh P
- Samarth G Vasist
- Navaneeth M

Fig. 6.10 Daily View of the User’s Calendar.



Fields	Values
Day	2021-04-17
Month	April
Year	2021
Weekday	Saturday
Week Start Day	2021-04-12
Week End Day	2021-04-18

Events

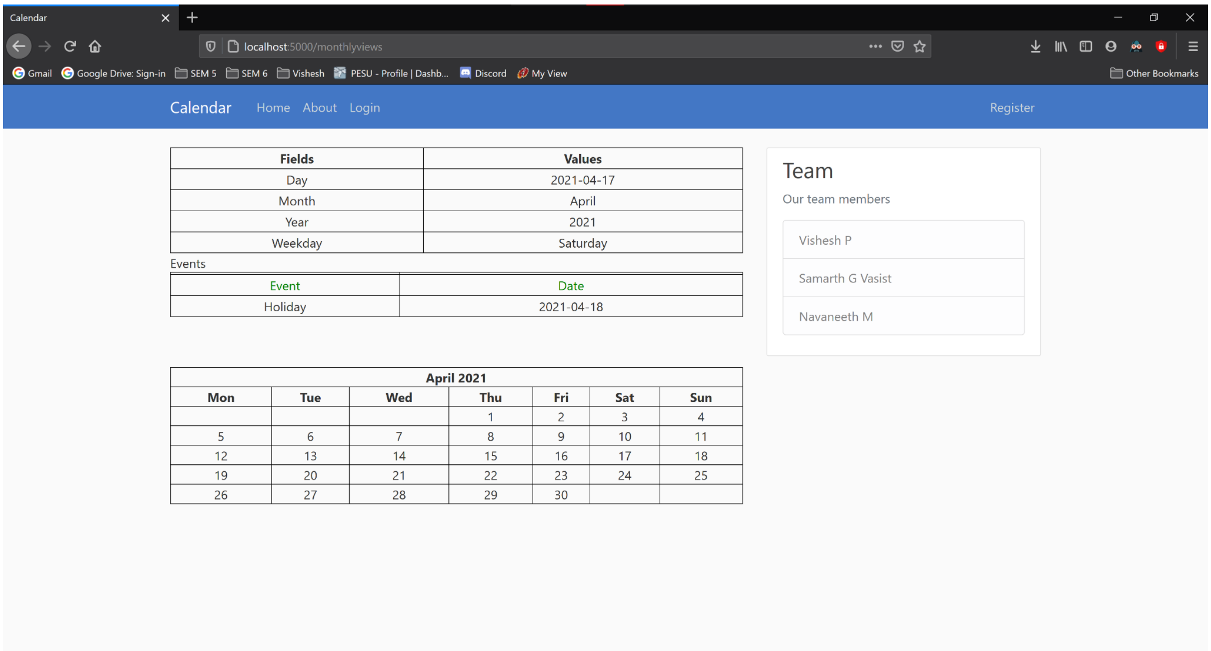
Event	Date
Holiday	2021-04-18

Team

Our team members

- Vishesh P
- Samarth G Vasist
- Navaneeth M

Fig. 6.11 Weekly View of the User’s Calendar



Fields	Values
Day	2021-04-17
Month	April
Year	2021
Weekday	Saturday

Events

Event	Date
Holiday	2021-04-18

April 2021

Mon	Tue	Wed	Thu	Fri	Sat	Sun
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30		

Team

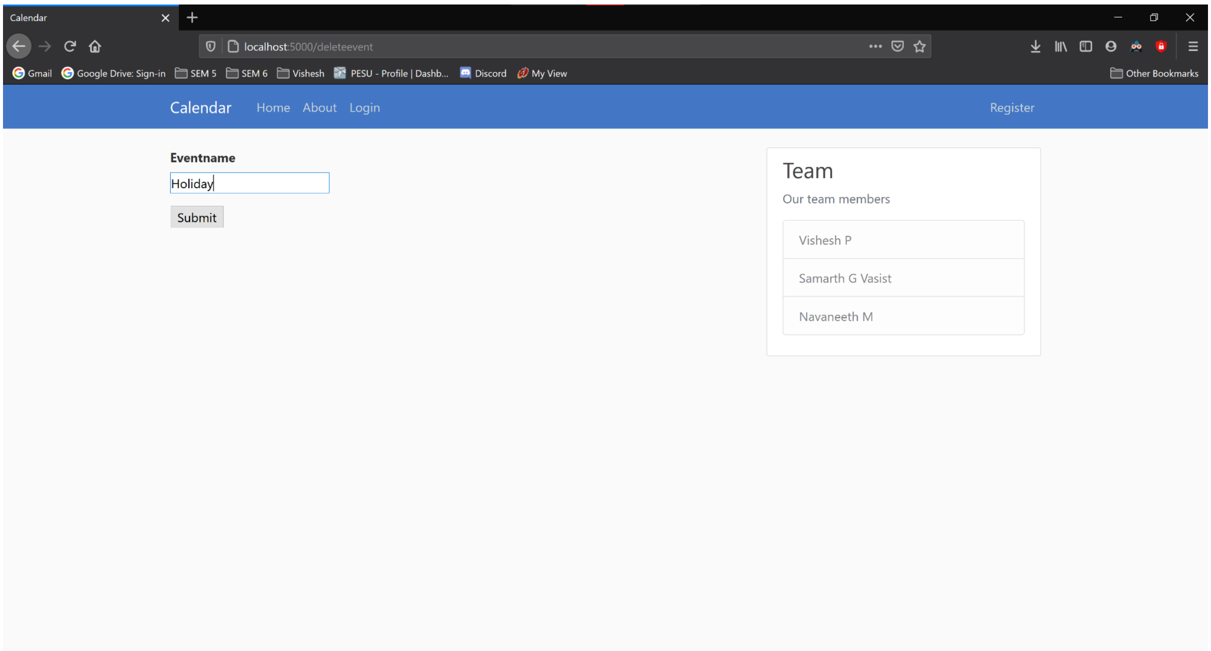
Our team members

Vishesh P

Samarth G Vasist

Navaneeth M

Fig. 6.12 Monthly View of the User’s Calendar



Eventname

Holiday

Team

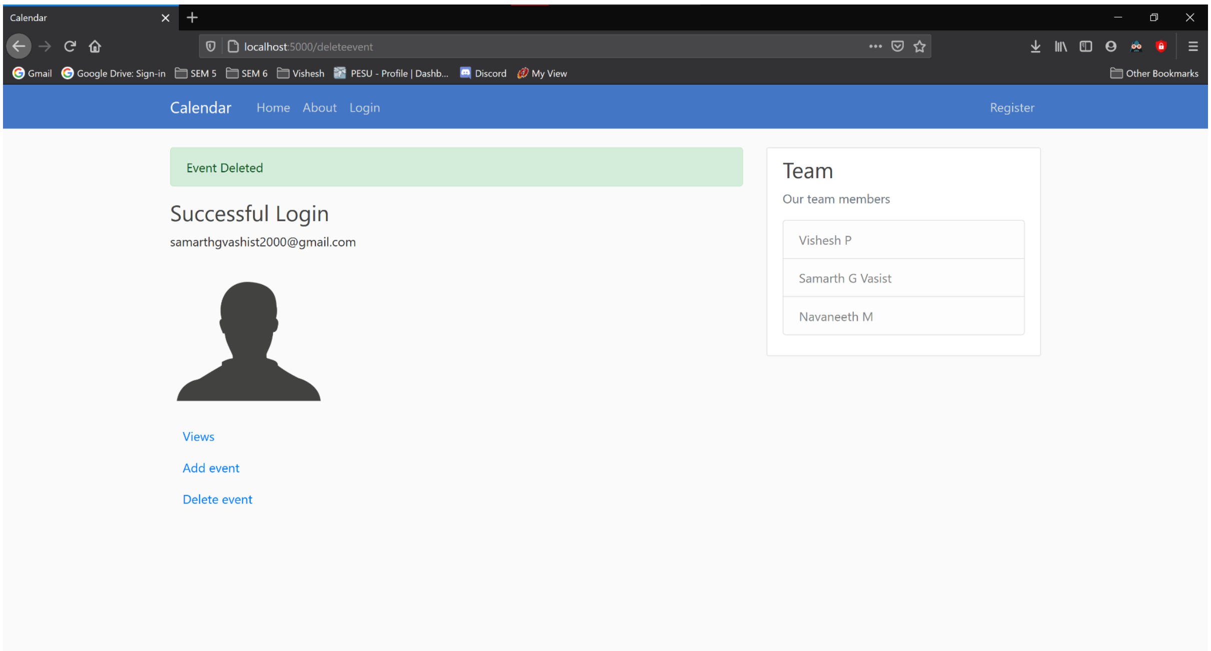
Our team members

Vishesh P

Samarth G Vasist

Navaneeth M

Fig. 6.13 Form input to delete any event in the User’s Calendar



Event Deleted

Successful Login

samarthgvashist2000@gmail.com

Team

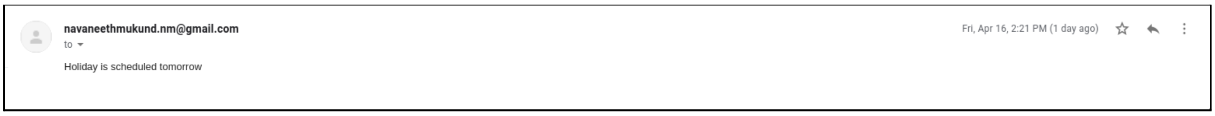
Our team members

Vishesh P

Samarth G Vasist

Navaneeth M

Fig. 6.14 Success message displayed on screen upon the deletion of an event



navaneethmukund.nm@gmail.com

to

Fri, Apr 16, 2:21 PM (1 day ago)

Holiday is scheduled tomorrow

Fig. 6.15 Email Reminder sent to the User’s Account (samarthgvashist2000@gmail.com)