SOFTWARE REQUIREMENT SPECIFICATION

For

Calender scheduler application

Prepared by:-

JAYASHRI V
KEERTHIGA S
MANTHARA G
GOPIKA M

Academic year : 2023-2024

1.Introduction:

1.1 Purpose:

A calendar app is a scheduling tool. It allows you to plan out your day by creating blocks in a calendar. Business meetings, appointments, events, daily reminders, work blocks, and more can be scheduled and organized through a calendar app.Calendar allows you to focus your view in the four obvious ways—Day, Week, Month, and Year.

A calendar scheduler serves the essential purpose of helping individuals and groups efficiently manage their time by organizing appointments, tasks, and events, facilitating effective time allocation, task prioritization, and deadline tracking, while also offering reminders and enabling collaboration for streamlined planning and coordination.

1.2 Document conventions:

- ➢ Entire document should be justified.
- ➢ Convention for Main Title
  - Font Face: Times New Roman
  - Font Style: Bold
  - Font Size: 14
- ➢ Convention for subtitle:
  - Font Face: Times New Roman
  - Font style: Bold
  - Font size: 12
- ➢ Convention for body:
  - Font Face:Times New Roman
  - Font Size:12

1.3 Scope of Development project:

Calendar scheduler is the ability to create a common calendar that others can use for scheduling an appointment or scheduling some time. As an example, take a small business looking to boost their online exposure by creating a scheduling app for users to book and pay for a service ahead of arriving.

Calendars are useful tools for keeping track of upcoming meetings, deadlines, and milestones. They can help you visualize your schedule and remind you of important events, such as holidays and vacation time.

It’s no wonder that people often have a variety of calendar tools to choose from, including everything from a paper calendar on their office wall to a calendar management tool, such as Calendly.

The problem is that people too often manage multiple calendars at once. When these calendars aren’t integrated with your work and synchronized with each other, this can lead to mass confusion, headaches, and missed deadline.

1.4 Definitions,Acronyms and Abbreviations:

Front end:

Node js	- platform runtime environment
React js	- flexible JavaScript library
HTML	- HyperText Markup Language
CSS	- Cascading Style Sheet
Javascript	- scripting language for creating dynamic web page content

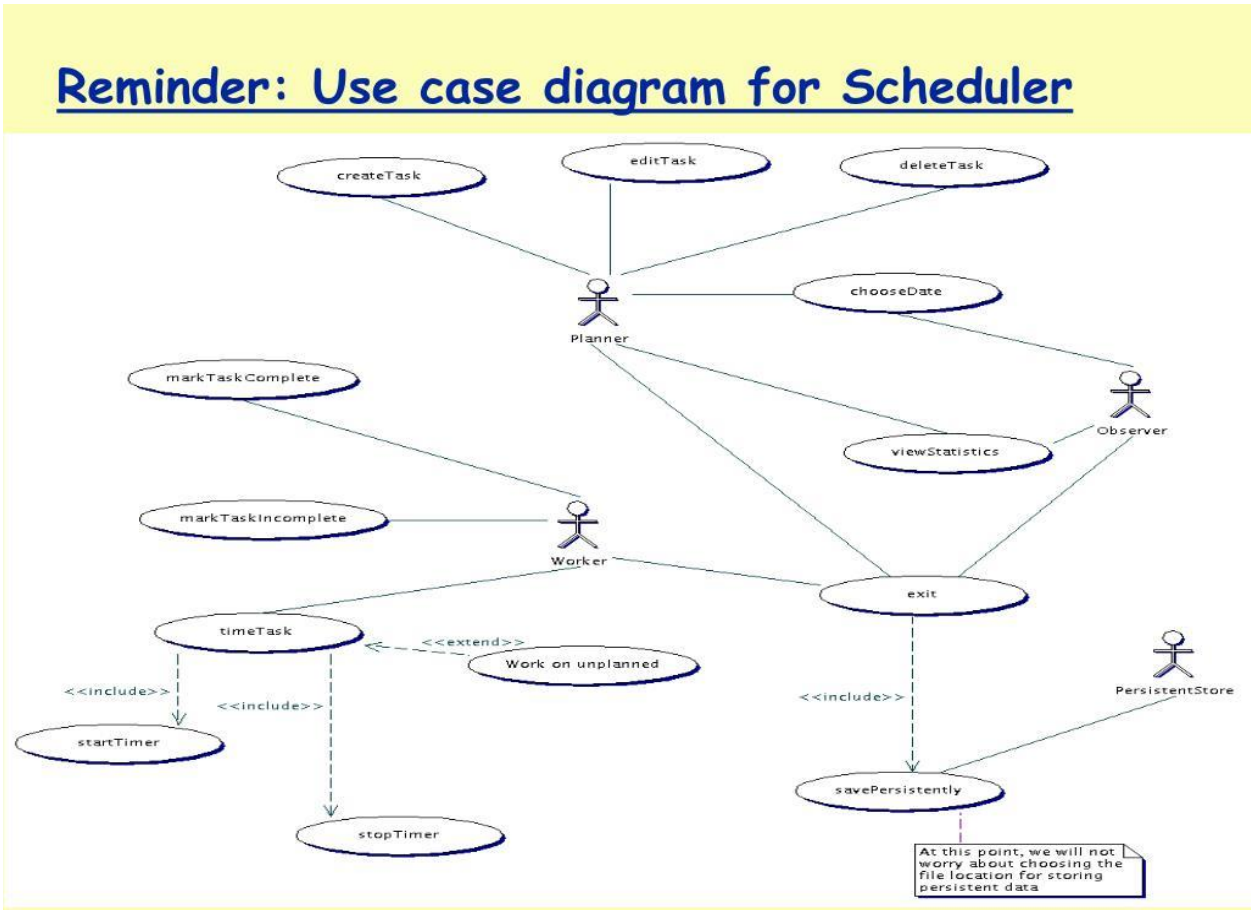
Back end:

Java - Platform Independence
SQL - Structured Query Language

1.5 References:

2.Overall Descriptions

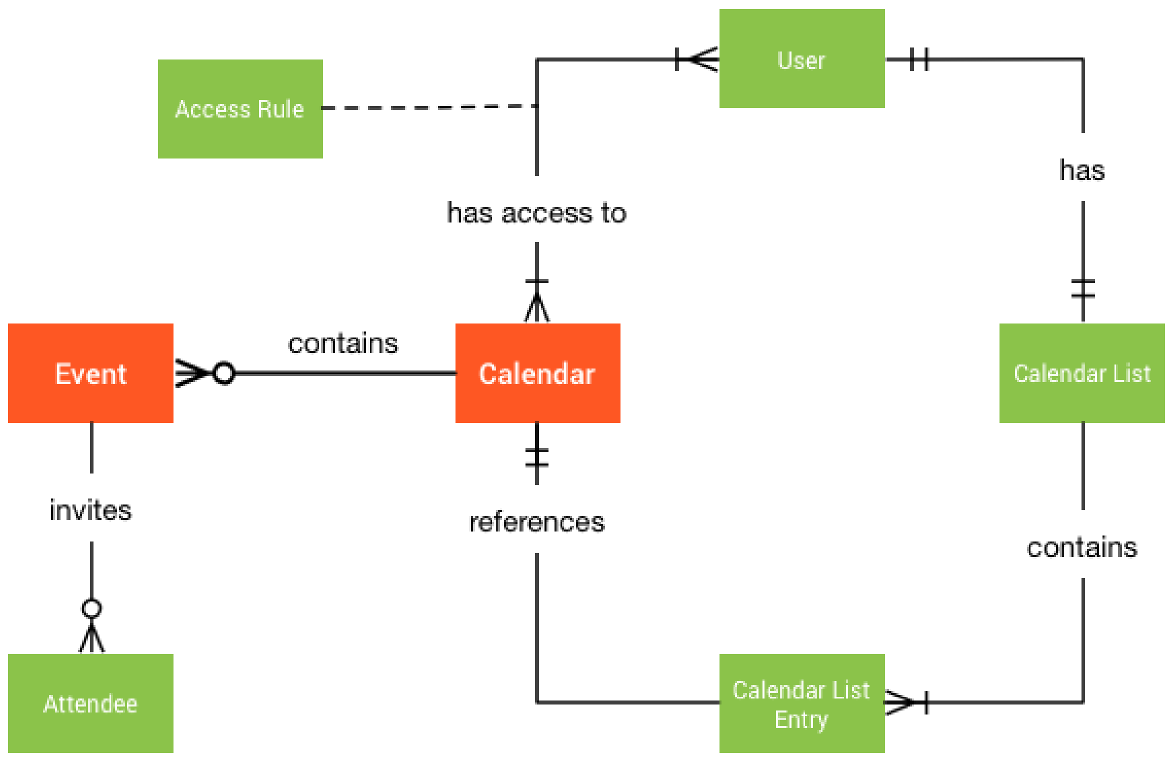
2.1 Product perspective:



At this point, we will not worry about choosing the file location for storing persistent data

2.2 Product function:

Entity Relationship for calender scheduler application



Creating a calendar scheduler application involves designing a wide range of functions to handle various tasks, from event creation and editing to scheduling and notifications. Below, I'll outline some key functions that you might need in your calendar scheduler application:

1. Create Event Function:
   - Allows users to create new events with details such as title, date, time, location, and description.

2. Edit Event Function:
   - Enables users to edit existing events, including changing details like date, time, or location.

3. Delete Event Function:
   - Allows users to delete events they no longer need.

4. View Calendar Function:
   - Displays the user's calendar with a view of events for a selected day, week, or month.

5. **Schedule Meeting Function**:
   - Provides a way to schedule meetings by checking the availability of participants and finding a suitable time slot.

6. Invite Participants Function:
   - Allows users to invite other users to their events or meetings and sends them notifications.

7. Set Reminders Function:
   - Lets users set reminders for events, with options for notifications via email, push notifications, or SMS.

8. Search Events Function:
   - Allows users to search for specific events based on keywords, date range, or other criteria.

9. Sync with External Calendars Function:
   - Integrates with external calendar services like Google Calendar, Outlook, or Apple Calendar to sync events and schedules.

10. Color Coding Function:
    - Provides the ability to color-code events for better organization and visualization.

11. User Authentication Function:
    - Implements user authentication and authorization to ensure data privacy and security.

12. Share Calendar Function:

- Enables users to share their calendars with others, specifying different levels of access (view-only, edit, etc.).

13. Notification Function:
    - Sends notifications for upcoming events, meeting invitations, and reminders.

14. Time Zone Support Function:
    - Handles events and meetings across different time zones accurately.

15. Recurring Events Function:
    - Supports the creation of recurring events like daily, weekly, or monthly appointments.

16. Export/Import Calendar Function:
    - Allows users to export their calendar data to a file or import data from external sources.

17. Weather Integration Function:
    - Provides weather forecasts for event locations to help users plan accordingly.

18. Data Backup and Recovery Function:
    - Offers regular data backups and a mechanism for users to recover lost data.

19. Analytics and Insights Function:
    - Provides users with insights into their scheduling habits and time management.

20. Mobile App Functionality:
    - Develops a mobile app version of the calendar scheduler for on-the-go access.

21. Desktop/Desktop Web App Functionality:
    - Creates a desktop or web-based version of the application for more extensive functionality.

Remember to design the user interface and user experience (UI/UX) to make these functions easily accessible and intuitive for users. Additionally, thorough testing and security measures are crucial to ensure a reliable and secure calendar scheduler application.

2.3 User Classes and Characteristics:

User classes and characteristics are essential considerations in the design and development of various products and services, including software applications, websites, and physical products. Understanding the different user classes and their characteristics helps designers and developers tailor their offerings to meet the diverse needs and preferences of their target audience. Here are some common user classes and their characteristics:

1. End Users:
   - End users are the primary users of a product or service.
   - Characteristics:
     - Vary widely in age, technical proficiency, and experience.
     - May have limited knowledge of the product or technology.
     - Seek ease of use, simplicity, and intuitive interfaces.
     - Value reliability and stability.

2. Administrators:
   - Administrators are responsible for managing and configuring a product or system.
   - Characteristics:
     - Have in-depth knowledge and expertise in the product or system.
     - Need access to advanced features and settings.
     - Require efficient tools for configuration and troubleshooting.

3. Developers:
   - Developers use a product or platform to create or customize solutions.
   - Characteristics:
     - Proficient in programming languages and technical documentation.
     - Seek robust APIs, developer tools, and extensibility.
     - Value comprehensive documentation and community support.

4. Managers and Decision-Makers:

- Managers and decision-makers make strategic choices related to the product or service.
- Characteristics:
  - Focus on the product's impact on the organization's goals.
  - Need clear reports, analytics, and ROI information.
  - May not be deeply technical but require insights to make informed decisions.

5. Customers or Clients:
   - Customers or clients are those who purchase or use the product or service.
   - Characteristics:
     - Seek value, affordability, and reliability.
     - May have specific requirements or expectations.
     - May require customer support and communication channels.

6. Guest or Anonymous Users:
   - Guest or anonymous users interact with a product or service without creating an account.
   - Characteristics:
     - May have limited access to features or content.
     - May seek quick access and a seamless user experience.
     - Privacy and security considerations are important.

7. Power Users:
   - Power users are advanced users who utilize the product's full potential.
   - Characteristics:
     - Embrace complexity and advanced features.
     - May explore and experiment with the product extensively.
     - Appreciate keyboard shortcuts, advanced settings, and customization options.

8. Accessibility Users:
   - Accessibility users have diverse needs related to disabilities.
   - Characteristics:
     - May require screen readers, voice commands, or other assistive technologies.
     - Value products and services that prioritize accessibility and inclusivity.

9. International Users:

- International users come from diverse cultural backgrounds and languages.
- Characteristics:
  - May require multilingual support and localization.
  - Seek culturally sensitive content and design.

10. Mobile Users:
    - Mobile users access products and services on smartphones and tablets.
    - Characteristics:
      - Value responsive design and mobile-friendly interfaces.
      - May have limited screen space and slower network connections.

Understanding the characteristics and needs of these user classes is crucial for creating user-centric designs and ensuring a positive user experience. User research, personas, and usability testing can help refine products and services to meet the expectations of each user class effectively.

2.4 Operating Environment :

For a calendar scheduler application, the operating environment is crucial to ensure the application can meet user expectations and function reliably. Here's a more specific operating environment tailored to a calendar scheduler application:

1. Operating Systems:
   - Support for multiple operating systems such as Windows, macOS, and Linux for desktop versions.
   - Compatibility with iOS and Android for mobile versions.

2. Hardware Requirements:
   - Specify minimum hardware requirements for devices running the application, including CPU, RAM, and storage.

3. Web Browsers:

- Ensure compatibility with popular web browsers like Chrome, Firefox, Safari, and Edge for web-based versions.

4. Network Environment:
   - Handle various network conditions, including slow or intermittent internet connections for web and mobile users.
   - Implement offline capabilities for mobile apps.

5. Database Systems:
   - Use a reliable database system for storing calendar data, like MySQL, PostgreSQL, or a cloud-based solution.

6. Security Considerations:
   - Implement robust security measures to protect user data and prevent unauthorized access.
   - Encrypt data both in transit and at rest.
   - Secure user authentication and authorization.

7. API Integrations:
   - Integrate with external calendar services like Google Calendar, Outlook Calendar, and Apple Calendar.
   - Utilize APIs for location services, weather forecasts, and notification delivery.

8. Mobile Platforms:
   - Develop mobile apps for iOS and Android, considering platform-specific design guidelines and requirements.

9. Localization and Internationalization:
   - Support multiple languages, date and time formats, and regional holidays.

10. Scalability and Performance:
    - Design for scalability to handle a large number of users and events.
    - Optimize performance to ensure responsive event creation and scheduling.

11. User Access and Permissions:

- Implement user roles and permissions to control access to calendar data.
- Define sharing settings for users to collaborate on events and schedules.

12. Backup and Disaster Recovery:
    - Regularly back up user data and implement a disaster recovery plan to prevent data loss.

13. Notifications and Reminders:
    - Ensure timely delivery of notifications via email, push notifications, or SMS.
    - Handle different notification preferences for users.

14. Compliance and Privacy:
    - Adhere to data privacy regulations like GDPR and HIPAA if applicable.
    - Clearly communicate privacy policies to users.

15. Environmental Sustainability:
    - Consider energy-efficient hosting options and eco-friendly practices in data center choices.

By carefully considering these operating environment factors, you can create a calendar scheduler application that meets user needs, operates reliably, and maintains data security and privacy. Additionally, regular updates and maintenance are crucial to adapt to changing technologies and user expectations.

2.5 Assumptions and Dependencies:

In the development of our calendar scheduler application, several key assumptions and dependencies have been identified. We assume that users will have access to compatible devices and stable internet connections, and that third-party calendar service APIs will remain available for integration. Additionally, the successful deployment of the application relies on cloud-based hosting services, ongoing software updates, and adherence to data privacy regulations. These assumptions and dependencies are integral to the application's functionality and must be carefully managed throughout its lifecycle to ensure a seamless user experience and system reliability.

2.6 Requirement :

Requirements for a calendar scheduler application are the essential criteria, features, and functionalities that must be met to satisfy user needs and expectations. Here is a high-level overview of requirements for such an application:

1. User Registration and Authentication:
   - Users should be able to create accounts and log in securely.
   - Support for password resets and multi-factor authentication is required.

2. Event Creation and Editing:
   - Users should be able to create, edit, and delete events with details such as title, date, time, location, description, and recurrence options.

3. Calendar Views:
   - Provide various views, including daily, weekly, and monthly calendars, to visualize events.

4. Notifications and Reminders:
   - Users should receive notifications and reminders for upcoming events via email, push notifications, or SMS.

5. User Roles and Permissions:
   - Implement user roles (e.g., admin, regular user) and permission settings for event access and sharing.

6. Integration with External Calendars:
   - Allow users to sync and import events from external calendar services like Google Calendar, Outlook, and Apple Calendar.

7. Search and Filter Functionality:
   - Provide a search feature for users to find specific events based on keywords, date ranges, or other criteria.
   - Enable filtering by categories or tags.

8. User-Friendly UI/UX:
   - Create an intuitive and responsive user interface for seamless navigation and ease of use.

9. Compatibility and Access:
   - Ensure cross-platform compatibility, including web, mobile (iOS, Android), and desktop (Windows, macOS, Linux).

10. Scalability and Performance:
    - Design for scalability to accommodate a growing number of users and events.
    - Optimize performance for responsive event creation and scheduling.

11. Security and Privacy:
    - Implement robust security measures, including data encryption, secure authentication, and access controls.
    - Comply with data privacy regulations (e.g., GDPR) and clearly communicate privacy policies to users.

12. Backup and Data Recovery:
    - Regularly back up user data and establish a disaster recovery plan to prevent data loss.

13. Localization and Internationalization:
    - Support multiple languages, date formats, and regional holidays.
    - Consider cultural differences in design and content.

14. Accessibility:
    - Ensure accessibility for users with disabilities by following accessibility standards (e.g., WCAG).

15. Offline Functionality:
    - Enable users to access and modify their calendars when offline, with changes syncing when connectivity is restored.

These requirements lay the foundation for a comprehensive calendar scheduler application that meets the diverse needs of users while maintaining security, privacy, and reliability. Detailed documentation and ongoing communication with stakeholders are crucial to successfully implement and evolve these requirements throughout the application's lifecycle.

2.7 Data Requirement:

The data requirements for a calendar scheduler application encompass the need to capture, store, and manage various types of information, including user profiles, event details (e.g., title, date, time, location, description), user preferences, access permissions, synchronization data for external calendars, notification settings, and event-related metadata. Additionally, the system must maintain historical event data, backups, and audit logs for security and accountability purposes. Data storage should be scalable and secure, with encryption measures to protect user privacy and compliance with applicable data protection regulations such as GDPR or HIPAA, if relevant. Efficient indexing and retrieval mechanisms are necessary to support search, filtering, and calendar views, ensuring that users can access and manage their calendar data effectively.

3. External Interface Requirement:

The calendar scheduler application requires a range of external interfaces, including user-friendly cross-platform interfaces for web and mobile, third-party calendar service integrations for event synchronization, location and weather forecast services for event details, notification services for reminders, developer APIs for extensibility, user support channels for assistance, import/export interfaces for data interchange, data backup and recovery mechanisms, analytics and reporting capabilities, compliance interfaces for regulatory requirements, payment gateways for premium features, social media sharing options, accessibility standards, and external documentation and help resources to ensure a seamless and comprehensive user experience.

4. System Features:

The system features of a calendar scheduler application encompass a set of functionalities and capabilities that provide users with the tools to efficiently manage their schedules and events. Here are some key system features:

1. Event Creation and Editing:
   - Users can easily create, edit, and delete events with details like title, date, time, location, description, and recurrence settings.

2. Multiple Calendar Views:
   - Offer various calendar views, including daily, weekly, and monthly, for users to visualize and manage their schedules.

3. Notifications and Reminders:
   - Send notifications and reminders to users via email, push notifications, or SMS for upcoming events.

4.User Authentication and Authorization:
   - Ensure secure user authentication and role-based access control to safeguard calendar data.

5. Integration with External Calendars:
   - Enable users to sync and import events from external calendar services like Google Calendar, Outlook, and Apple Calendar.

6. Event Sharing and Collaboration:
   - Allow users to share events with others and set permissions for viewing or editing, facilitating collaboration.

7. Search and Filter Functionality:
   - Provide robust search and filtering options to help users find specific events quickly.

8. Time Zone Support:
   - Handle events and scheduling across different time zones accurately.

9. Recurrence Patterns:
   - Support various recurrence patterns, such as daily, weekly, monthly, and custom schedules.

10. Backup and Data Recovery:
    - Implement data backup and recovery mechanisms to prevent data loss.

11. Customization and Personalization:
    - Allow users to customize calendar appearance, color-coding, and event categories.

12. Offline Functionality:
    - Enable users to access and modify their calendars offline, with changes syncing when connectivity is restored.

13. Location Services Integration:
    - Utilize location-based services and maps for event locations and directions.

14. Weather Forecasts:
    - Provide weather forecasts for event locations to assist users in planning.

15. User Preferences and Settings:
    - Offer a range of user preferences and settings for notification preferences, time formats, and regional settings.

16. Accessibility Features:
    - Ensure the application is accessible to users with disabilities by adhering to accessibility standards (e.g., WCAG).

17. Analytics and Insights:
    - Incorporate analytics tools to provide users with insights into their scheduling habits and time management.

18. Support and Help Resources:
    - Offer user support through various channels and maintain comprehensive help resources, including FAQs and user guides.

19. Mobile Apps:

- Develop mobile applications for iOS and Android platforms with responsive design and offline capabilities.

20. Desktop/Desktop Web App:
    - Create a desktop or web-based version of the application for extended functionality.

These system features collectively provide users with a powerful calendar scheduling tool, ensuring efficient event management, collaboration, and a seamless user experience across different platforms and devices.

5. Other Non-functional Requirements:

5.1 Performance Requirement :

Performance requirements for the calendar scheduler application dictate that it should load within a specified timeframe, ideally under two seconds, and respond swiftly to user interactions. The application must be capable of supporting a significant number of users and events simultaneously, with a target of at least 1000 concurrent users without substantial degradation in response times. Load testing should be regularly conducted to ensure the system can handle peak loads efficiently, and performance optimizations should be applied as needed to maintain responsive user experiences, particularly when viewing, editing, or searching for events.

5.2 Safety Requirement:

Safety requirements for the calendar scheduler application are focused on ensuring user data privacy and the prevention of unauthorized access. This includes robust data encryption both in transit and at rest to safeguard user information. Additionally, the application should implement secure authentication mechanisms and access controls to prevent unauthorized users from accessing or modifying sensitive calendar data. Compliance with relevant data protection regulations, such as GDPR or HIPAA, should be a key consideration to ensure the safety and confidentiality of user data. Regular security audits and prompt patching of vulnerabilities are essential to maintain a safe environment for users.

5.3 Security Requirement:

The security requirements for the calendar scheduler application include enforcing strong authentication, data encryption in transit and at rest, regular vulnerability assessments, compliance with data privacy regulations, timely security patching, comprehensive audit logging and monitoring, secure file handling, staff security training, incident response planning, and ensuring the security of third-party components. These measures collectively aim to protect user data, prevent unauthorized access, and maintain a secure and reliable system.

5.4 Requirement attributes:

Requirement attributes for the calendar scheduler application provide additional context and details about each requirement. These attributes include priority (e.g., high, medium, low), source (e.g., user, regulatory body), stability (e.g., stable, evolving), and dependencies (e.g., integration with external services). Assigning these attributes helps prioritize and manage requirements effectively, ensuring that critical functionality is developed first, compliance with regulations is maintained, and dependencies on external systems are clearly understood and accounted for throughout the development process.

5.5 Business Rules:

The business rules for the calendar scheduler application define the logic and guidelines governing how users interact with the system. These rules encompass event creation constraints (e.g., maximum event duration, minimum notice period for recurring events), access control policies (e.g., who can view or edit specific events), notification preferences (e.g., user-defined notification methods and timing), and data retention policies (e.g., duration of event history). These rules are crucial for ensuring consistent user experiences, data integrity, and compliance with organizational policies and user expectations.

5.6 User Requirements:

User requirements for the calendar scheduler application encompass the specific needs and expectations of the application's target user base. These requirements include the ability to create and manage events with ease, customize calendar views, receive timely notifications and reminders, collaborate with others by sharing events, search for events efficiently, and access the application seamlessly across various devices and platforms. Users also expect a user-friendly interface, robust security measures to protect their data, and compliance with privacy regulations to ensure the confidentiality of their information. Meeting these user requirements is essential to providing a satisfying and effective scheduling experience.

6. Other Requirements

6.1 Data and Category Requirement:

The data and category requirements for the calendar scheduler application involve the management of event data, including event details (titles, descriptions, times, dates, locations, recurrence), event categories and tags for organization, event attachments, location information, user preferences (time zones, formats), ownership and sharing settings for events, data history for auditing and versioning, and user profiles with essential user information. These requirements ensure efficient event organization, customization, and user personalization within the application.

6.2 Appendix:

The appendix in the documentation for the calendar scheduler application serves as a repository for supplementary information, including a glossary of terms, references to external resources, sample event data, technical diagrams, user interface mockups, regulatory compliance documentation, details about third-party service integrations, and any specific requirements related to the appendix itself. This appendix enhances the document's comprehensibility and provides readers with additional context and references to support their understanding of the application's development and functionality.

6.3 Glossary:

A glossary in software documentation is a valuable section that provides definitions and explanations for key terms, acronyms, and technical jargon used throughout the document. It helps ensure clarity and consistency in communication by offering a quick reference for readers who may encounter unfamiliar terminology. In the context of a calendar scheduler application, the glossary might include terms such as:

1. Event: A scheduled activity or appointment with specific details, including date, time, location, and description.

2. Recurrence: The pattern by which events repeat, such as daily, weekly, monthly, or custom schedules.

3. Category/Tag: A label or identifier assigned to events for organizational purposes.

4. Notification: A message or alert sent to users to remind them of upcoming events.

5. User Authentication: The process of verifying the identity of a user, typically through a username and password.

6. API (Application Programming Interface): A set of rules and protocols that allow different software applications to communicate with each other.

7.User Interface (UI): The visual elements and controls through which users interact with the application.

8. Data Encryption: The process of converting data into a code to prevent unauthorized access.

9. Access Control: A security mechanism that restricts or grants user access to specific features or data.

10. Compliance: Adherence to legal or regulatory requirements, such as data privacy regulations (e.g., GDPR) or industry-specific standards.

11. Backup and Recovery: Procedures for creating copies of data and restoring it in case of data loss or system failures.

12. Load Testing: Evaluating the performance of the application under various levels of user activity and load.

13. Vulnerability Assessment: The process of identifying and addressing security vulnerabilities in the system.

14. Incident Respons: A plan for handling and mitigating security incidents or breaches.

15. Accessibility: Ensuring that the application can be used by individuals with disabilities.

16. Data Privacy: Protection of user data and compliance with privacy regulations.

17Integration: The process of connecting the application with external services or systems.

18. Mobile App: A software application designed to run on mobile devices, such as smartphones and tablets.

19. Cross-Platform: Compatibility across different operating systems or platforms.

20. User Preferences: Customizable settings that users can configure to tailor the application to their needs.

21. User Profile: Information about a user stored within the system, typically including username, email, and preferences.

22. Notification Preferences: User-configurable settings for receiving event reminders and notifications.

23. Data History: A record of changes made to event data over time, allowing for auditing and version control.

Including a glossary in your documentation helps readers, including team members and stakeholders, better understand the terminology used in the context of the calendar scheduler application, promoting clear and effective communication.

6.4 Class Diagram:

A class diagram is a visual representation in software engineering that illustrates the structure and relationships between classes or objects in a system. In the context of a calendar scheduler application, a class diagram might depict the various classes or entities within the application and how they interact. Some classes commonly found in such a diagram could include:

1. User: Representing user profiles with attributes like username, email, and preferences.

2. Event: Modeling event details such as title, description, date, time, location, and recurrence.

3. Category/Tag: Describing categories or tags used for event organization.

4. Notification: Representing the sending and receiving of event notifications.

5. Calendar: Depicting the concept of calendars, which may contain multiple events and be associated with users.

6. Security: Illustrating classes responsible for user authentication, authorization, and data encryption.

7. Integration: Representing classes responsible for integrating with external services like third-party calendar providers or location services.

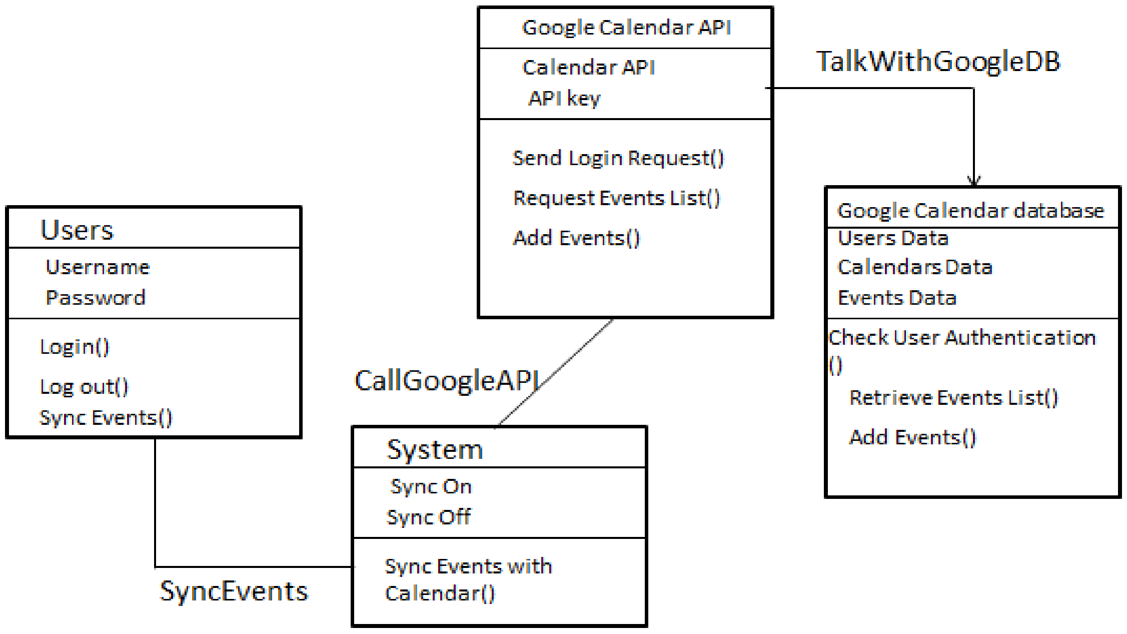
8. Settings: Describing classes that manage user preferences and application settings.

9. Audit Log: Modeling classes involved in recording and storing event history and user actions for auditing purposes.

10. Access Control: Depicting classes responsible for managing user access permissions.

11.Data Storage: Representing classes related to database storage and retrieval.

12. APIs: Illustrating classes responsible for providing or consuming APIs for external services.


Google Calendar API
Calendar API
API key
Send Login Request()
Request Events List()
Add Events()
TalkWithGoogleDB
Google Calendar database
Users Data
Calendars Data
Events Data
Check User Authentication ()
Retrieve Events List()
Add Events()
Users
Username
Password
Login()
Log out()
Sync Events()
CallGoogleAPI
System
Sync On
Sync Off
Sync Events with Calendar()
SyncEvents