Kaaj List

Kamrun Mim

Application Problem Statement:

We are looking to build an application that will help students facilitate their time and task management throughout the academic year through a digital planner application. This will be a schedule management resource students can customize to their class schedule, larger projects, reminders, and tasks. Currently, planners tend to be physical books that come in a variety of styles and designs however, this leaves limited customization options for the user. The design styles in which physical planners are created vary from plain to complex depending on your search for a specific style that suits your work style. Additionally, users tend to utilize the different functions of multiple applications to keep track of their schedules, and tasks like Google Calendar and Google Tasks making it difficult to maintain and manage these applications separately. Although there are digital planner applications like Notion, these applications are not accessible to everyday users because they're too complex to establish and maintain daily. For example, the setup of one's notion can be so overwhelming and complex that it defeats the purpose of using a digital planner. Lastly, regardless of digital or physical planners, we've noticed a lack of support available in all planners so we look forward to including a supportive environment for students by incorporating interactive sticky notes peers can send to your digital planner. Students shall be able to access their digital planner through any device as they will be able to link their account to their G-mail account.

Our digital planner application will provide a user-friendly interface for organizing and accessing personal schedules. The system should allow users to create, edit, prioritize, and categorize tasks and events, set reminders, and receive notifications. Additionally, the solution should support the ability to sync data across multiple devices seamlessly and securely, ensuring data integrity and confidentiality. The aim is to develop a robust and scalable software solution that enhances productivity and organization for individuals managing busy schedules in both personal and professional contexts.

Requirements:

1) Our Kaaj List application shall include skeuomorphism–interface objects that mimic their real-world counterparts in how they appear and/or how the user can interact with them. The application shall have an appealing and customizable GUI, making for a more engaging experience.
    a) The user interface shall resemble a bulletin board.
    b) The user should be able to arrange/rearrange objects (stickers, calendar, clock, etc.) on their bulletin board in whatever manner they choose.
    c) The user shall add customizable objects to their bulletin board such as sticky notes and stickers, encouraging the user to use their planner and creating a less mundane experience.
    d) The user shall also be able to delete these objects from their bulletin board.
    e) The font shall have a handwritten look.
    f) There will be a clock that resembles a real-life clock.
1) Our application must have a calendar that the user shall interact with.
    a) The user shall add events to the calendar
    b) The user shall delete events from the calendar.
    c) The user must have the option to receive notifications about scheduled events.
2) The user should have the ability to send their friends messages on sticky notes. This will be implemented by email. The user will write a message to their friend on a sticky note, and enter the email of their friend. The sticky note will then be sent to the friend's bulletin board. This will also promote engagement in both the user and the friend and keep them motivated in their studies/homework, by receiving motivating or funny messages from friends.

3) The user shall have the ability to manage tasks using task lists, making assignments/homework/etc. easy to organize and complete.
    a) The user shall create as many task lists as they want (for example, a task list for each class they are in).
    b) And shall also have the ability to delete any list.
    c) The user shall add tasks,
    d) Mark tasks as completed,
    e) And delete tasks.
    f) There will also be an option to assign a task a due date.
    g) The user will be able to set reminders for tasks to be completed
4) The application should be able to be accessible from any device as it uses email to sync data from devices. This allows for accessibility.

Use Cases

- There is an actor called User, user is the average user for this app. They will be referred to in the following use case descriptions
- View Main Screen - The user will open the app, and upon the app being opened it will load to a main screen.
    - Upon the very first time the app is opened the user will initialize an acount. The app will then ask the user for permission to access the clock and calendar of the phone. This will then use the phones built in function as external actors that provide this information to the app.
    - After the first opening, and on any subsequent app usages the main screen shall load. This main screen calls upon information from many other objects within the application.
        - In order to display the date and time the app will call upon the date and time from the phone itself (external system actors). These will be displayed as a literal clock and calendar GUI elements
        - If there are any sticky notes (other objects) those will be displayed as well. Otherwise none will be displayed. Likewise if any messages from friends will appear as stick notes (of other colors?)
        - If there are any To-Do Lists (other objects) the most recent one will be displayed. Otherwise a blank To-Do will be displayed. Upon tapping on this they will begin the Add To-Do List use case.
        - The app will also pull from the (in app) Calendar of events, and if any of display indicators on the visual calendar GUI. Otherwise nothing will change.

    - Tapping on any of the GUI Elements that represent sections of the app will be able to be tapped on. These will be the first steps of most other use cases of the app
- Create/View task list - The user will be able to create a list of tasks to do
    - After tapping the GUI element on the main screen they will be brought to the To-Do List screen.
    - From here the User can simply view their to-do lists, stored somewhere in the app's memory as objects.
    - Somewhere on the screen there will be a button to press to add a new one. This will create a new blank To-Do list (a type of object) that will then be added to the top of all the other To-Do lists (it will be this list that will be called upon in the home screen)
    - Further action will be taken in the next Use-Case
- Add/Edit Task - The user will be able to add a task to one of the to-do lists they have created
    - For the sake of ease a To-Do Object will always display room for n+1 task within it.
    - Upon tapping this empty spot on any To-Do List a text prompt will open. From here the user can type their task into the list. This individual task is saved as its own object.
    - After this point the user is given the optional ability to attach a date to this task object. This date/task will then automatically be added to the in app calendar.
    - If the user wishes to edit some existing task they must simply tap on the text of the task to edit. This will bring up a text window with the previous text of the task to edit. If they wish to edit the date they can tap the date next to the task (or the empty space if there is no date)
- Mark Tasks as Completed/Delete Tasks - The user will be able mark any task in a To-Do list as completed or delete
    - For any individual task the user will be able to delete a task
    - For this they will need to long press (or right click) any task and an option to delete will appear.
    - Pressing this delete button will delete that task object, as well as removing its place in the To-Do list visually, and setting the size of the todo list -1
    - Next time the user returns to the Home Screen the data pulled will update with this change
- Add Sticky Notes - On the home screen the user will be able to add sticky notes for a bit of individual flare
    - On the home screen the user will be able to tap anywhere that isn't any of the other menu objects to place a sticky note
    - Upon tapping an area a new sticky note object will be created, immediately recording the location on screen where the user tapped
    - From there a text box will appear. The user will be able to type in a message.
    - Or alternatively upload a photo from their device, this is another actor. Or some combination of both
    - Once the user is satisfied they can press Enter to finish the Sticky Note.
    - It will now display on the home screen where the user placed it.
- Delete Sticky Note - At any time the user will also be able to delete any sticky notes they placed
    - This process is similar to the deletion of task from lists
    - Simply tap and hold/right click on a sticky note for a delete button to show up

    - Tapping this will delete that Sticky Note object from memory and remove it from the home screen
    - This one is pretty simple
- View Calendar - Being able to view the calendar feature on the app
    - By tapping the Calendar GUI element the calendar screen of the app will oppen
    - The app will, by pulling data from the phone itself, display a calendar accurate to the current month
    - The app will then pull the data from any of the to-do lists, and any tasks within them with a due date in that month will cause a marker to appear on that day in the calendar.
    - Similarly any Events, to be described soon, that were set to take place on days will populate those days. It will show the names (that it can fit) as well as the time of day these events will take place.
    - By tapping on any specific day the app will change screens to display a more detailed overview of just that day. From here the user will be able to add events.
    - Days themselves will be there own objects
    - Additionally if you instead tap on a specific event or to-do marker the app will display just the information for that specific object.
- Add Events to Schedule - Adding some kind of timed event to
    - On the screen for any specific date the user will be able to hit a gui element (such as a + sign) in the corner of the screen.
    - This will immediately create a blank Event object
    - Upon clicking this a text box will appear, similar to the tasks, and it will allow you to name (set that variable's value) to that name
    - Afterwords the user will be given the option to, optionally, set the time this event happens at. This is another variable that is part of the Event Object
- Delete Events From Schedule - The user can delete any event they've made
    - This one is similar to the previous two deletes
    - The user taps and holds/right clicks an event and a prompt to delete it appears
    - If they click that the object is removed from memory and all areas calling upon that object stop using it
- Notifications - The app will send notifications to the user to remind them of events/tasks
    - Once an event is made a function within it will schedule a notification
    - This notification will be the name variable of any given event or task
    - The time it will be scheduled is for either an hour before the time of an event, or the day of (at 12:01 am) the day of the event or task.
    - Once that time is reached, even the closed app, will be able to send this notification
- Add Email - The user will be able to add an email to sync tasks between different devices/backup (this use case is a lower priority)
    - Upon First entry into the app the user will be prompted to provide an email and password
    - Once they do this that information will be sent to a server (in our case probably a Mac Mini or something of the like)
    - Every time the user closes the app the app will attempt to send all of their events/to-do list objects to be backed up in the server, encrypted and tied to their email

    - Whenever the app is open the app will make a call to that server to check if the data is synchronized. If not the user will be given a prompting to either restore the backup, or erase it and back up what's stored locally
    - The exception to this is sticky notes, this will be explained in the next use case
    - If the backup is restored all aforementioned objects will be replaced with the backup versions
    - Otherwise the backup will be erased and a new backup will be created
- Send Sticky Note - The user will be able to send a sticky note to a friend's app (this use case is a lower priority)
    - The user will be given an option to, when tapping a sticky note, send it to a friend
    - If they select yes the user will be prompted to enter the friend's email
    - Once entered the app will communicate with the server and check if the email can be found
    - If not, the app will respond with an error message
    - If yes the app will add that sticky note to the backup to be restored next time
    - Once the friend's app is opened next, the sticky note will appear on their board. It will have the email of the user who sent it written on it, and will be a different color to differentiate it.
- When creating this I believe the first use cases to do are viewing the home screen, as well as viewing the calendar and task list screens. This is because of how many other uses cases rely on them. Without the home screen a user cannot get to any of the other screens, so it needs to be made first. After these are made we can build the other use cases, like all the Adds, Edits, Delete cases, on top of them. As well, for us the sticky notes, sending messages, and user backups are all the least important.

Use Case: Render Main Screen

CRC diagram:
Please see the attached picture

1. Used Cards
    a. Bulletin Board
    b. To-Do List
    c. User Profile
    d. Calendar
    e. Sticky Notes
2. Initially load in the app
3. First the app will call upon the Profile Class for the user information. This will inform further classes
4. Immediately after the app will call upon the Bulletin Board class

5. This will then display the bulletin board it will call upon the information provided by its collaborators.
6. First it will call upon the Calendar class to be able to then display the users calendar.
7. It will then display that calendar
8. It will repeat this process for the To-Do Lists
9. Then display the To-Do
10. Then, when calling upon the sticky notes classes they will check their own methods to determine what is their content and where should they place themselves
11. They will then also call on the Profile class to determine if they are an original or sent sticky notes, if sent they know to change their color as such
12. Finally the sticky notes will make themselves visible on the bulletin board.

Use Case View To-Do Lists:

1. Used Cards
    a. To-Do List
    b. Tasks
2. The user will load into the To-List Section of the App
3. Immediately, a/all current To-Do List class will be loaded
4. This list will call upon its collaborators, that being the Task objects in this list
5. When the user wants to view another todo list another List object will be loaded, and step 4 will be repeated again.

Use Case Create To-Do List:

1. Used Cards
    a. To-Do List
    b. Tasks
    c. Bulletin Board
2. When the user creates a To-Do list a new, empty, To-Do List Class will be called upon
3. This list will a variable known as "Name" that a user must complete before the To-Do List is saved
4. After the list itself is initially created it wil call upon a task collaborator by creating 1, blank, Task
5. This task will then be noted to be part of this initial list.
6. The new task list communicates back to the Bulletin Board Class to set this new list as the most recently edited one to be displayed.

Use Case Add Task:

1. Used Cards
    a. Task
    b. Task List
    c. Calendar
2. The user will hit a button to add a task to a given task list
3. The list itself will communicate with blank task item.

4. The previously blank task for this list will be filled in with the name of the list, and what index in the list is
5. Then the user will be told to enter text for what the task itself
6. Optionally, after that the user will enter a due date for the task
    a. If this occurs the task object will set up a communication to the Calendar object to display this task on the calendar
7. The task object will then expand the size of the List by 1, and create a new blank task object so that this process may be repeated

Use Case Add Sticky Note:

1. Used Cards
    a. Bulletin Board
    b. Sticky Note
    c. Profile
2. The user will tap somewhere on the bulletin board to create a sticky note
3. A new sticky note object will be created
4. Immediately upon its creation it will call to the BB collaborator to check what location it is stored at. Then it will store that information
5. Afterwards the sticky note will call to the Profile class to determine the email of the person who made it. It will also store this information
6. Afterwards the user will be given prompted to enter the text that goes on the sticky note
7. Once this is done the sticky note will store that information
8. Then the sticky note will be visible on the screen when the bulletin board is displayed.

Use Case View Calendar:

1. Used Cards:
    a. Tasks
    b. Calendar
    c. Events
    d. Days
2. The user will load into the calendar section of the app
3. When loading in the calendar object will call for information from 3 collaborators
4. It will pull the names and due dates of any Task objects to display on the calendar
5. It will call upon all the Day objects
6. From the day objects the Event objects will also be called for their name, date, and times
7. All of this information will be displayed on the calendar.

CRC Card: Bulletin Board

Class Name	**Bulletin Board**
Responsibilities	Collaborators
- Display Calendar - Display To-Do List - Display Sticky Note - Create Sticky Note - Delete Sticky Note - Display time	- To-Do Lists - Calendar - Profile - Sticky Note

Class Name	**Profile**
Responsibilities	Collaborators
- Store Email - Add Email - Store Name - Store Friends - Add Friend - Delete Friend	- Bulletin Board - Sticky Notes

Class Name	Calendar
Responsibilities	Collaborators
- Query Event - Create Event - Delete Event - Check Phone Calendar	- Events - To-Do List - Day - Bulletin Board

Class Name	**Day**
Responsibilities	Collaborators
- Create Event - Delete Event - Query Events Attached to Self	- To-Do Lis - Calendar - Event

Class Name	**Event**
Responsibilities	Collaborators
- Store Name of Event - Store Time of Event	- Calendar - Day

Class Name	**Sticky Note**
Responsibilities	Collaborators
- Track Creator - Store Text of Self - Store Location on Board - Send Self	- Bulletin Board - Profile

Class Name	**To-Do List**
Responsibilities	Collaborators
- Create Task - Delete Task - Keep Track of Tasks Attached to Self	- Tasks - Bulletin Board - Calendar

Class Name	**Task**
Responsibilities	Collaborators
- Store Name of Task - Store Time of Task - Store Name of List Attached To	- To-Do List