Software Requirements Specification

Team:
Modern Family 2031

Table of Contents

Table of Figures

1 Introduction

1.1 Purpose

This document is intended to clearly describe an overview of the requirements and specifications for a mobile application that facilitates family coordination. It contains detailed outlines of the three major use cases of this application in the form of various models and diagrams. Furthermore, this document contains implementation details that can help bring these use cases to fruition.

The intended audience of this document consists primarily of software developers who will be implementing the final product, and to some degree, users of the mobile application. This document is to be used as a standard guide or reference for the developers who will be implementing the final product as it highlights the requirements and specifications of the application from a technical perspective. Users may reference this document to understand how the application is intended to be used. The audience is assumed to have a basic knowledge of Unified Modeling Language (UML), state machines, domain models and use case diagrams.

1.2 Scope

The mobile application that is described by this document, entitled Modern Family 2031, is aimed at providing busy families in North America a means to stay connected, and coordinate time or resources while minimizing conflicts. Specifically, Modern Family 2031 assists the family members, who have accounts on the application, to easily organize family activities by providing a smart planning tool that integrates with each family member's personal calendar and makes suggestions based on the availabilities of all the family members. The application also provides a resource inventory, where family members can find and modify data about who, when, and why a shared family resource is occupied. Family members can make bookings or override requests for their desired resources to minimize conflicts that arise from shared usage. From the perspective of staying connected, the application allows the family members to collaborate on media playlists that support different media types such as music, videos, or pictures. The main benefit of Modern Family 2031 is to improve family bonding because all the features provided by this application emphasize a sense of shared responsibility among all members of the family. Also, considering approximately 86.1% of Canadian households own at least one smartphone in 2021 [2], and the platform that Modern Family 2031 relies on is mobile, the barrier to access the services is low. Due to the nature of the mobile platform, Modern Family 2031 can also bring the convenience of real-time updates as long as the mobile devices stay connected with the internet. On the other hand, Modern Family 2031 also has some boundaries since it requires most family members in a household to have accounts with it to make the aforementioned features effective. Moreover, people who lack technology savviness may have a steeper learning curve as first-time users of the application.

1.3 Acronyms, Abbreviations, Definitions, Notational Conventions

Abbreviations

Abbreviation	**Expansion**
API	Application Programming Interface
UI	User Interface
URL	Uniform Resource Locators
POV	Point of View
UML	Unified Modeling Language

Conventions

Convention	**Meaning**
Application	Also used to refer to our system
User	Sometimes used to refer to the “family member” (users of the system)
Host	Refers to a family member that initiates and plans a family activity
Creator	Refers to a family member that creates a new playlist
Invitees	Refers to family members invited to a family activity, excluding the family member hosting the activity (the Host)
Editors	Refers to family members who are invited to collaborate on a playlist, excluding the family member who created the playlist (the Creator)
“”	Text inside “” is used to refer to button names or other content on the UI
Underline	Something that is underlined represents a template statement. For example, the user stories follow this template: As a <user/role>, I want <something>, so that <benefit is achieved>
<...>	Something in <...> serves as a placeholder for some dynamic variables on the UI. For example, “<Family Member Name>” on the Menu Screen.
Italics	Denotes figure captions.
[#]	The # in the square brackets corresponds to the No. column in the UI sketches description table.

2 Overall Description

This section serves as an introduction to the system as a whole, starting with a high level overview in the form of a use case diagram, and then going into the details of the three main use cases through workflow models. Additionally, the user characteristics, general constraints, and assumptions and dependencies are details in this section as well.

2.1 Product Perspective

Modern Family 2031 works together with a third-party calendar service, an email service, and third-party media services (i.e. YouTube, Spotify, iTunes, etc.) to bring each use case to fruition. The major use cases that this application provides include planning a family activity, coordinating the usage of shared family resources, and collaborating on shared media playlists.

The use case diagram of Modern Family 2031 below showcases the interaction between Modern Family 2031 and its human and external actors.

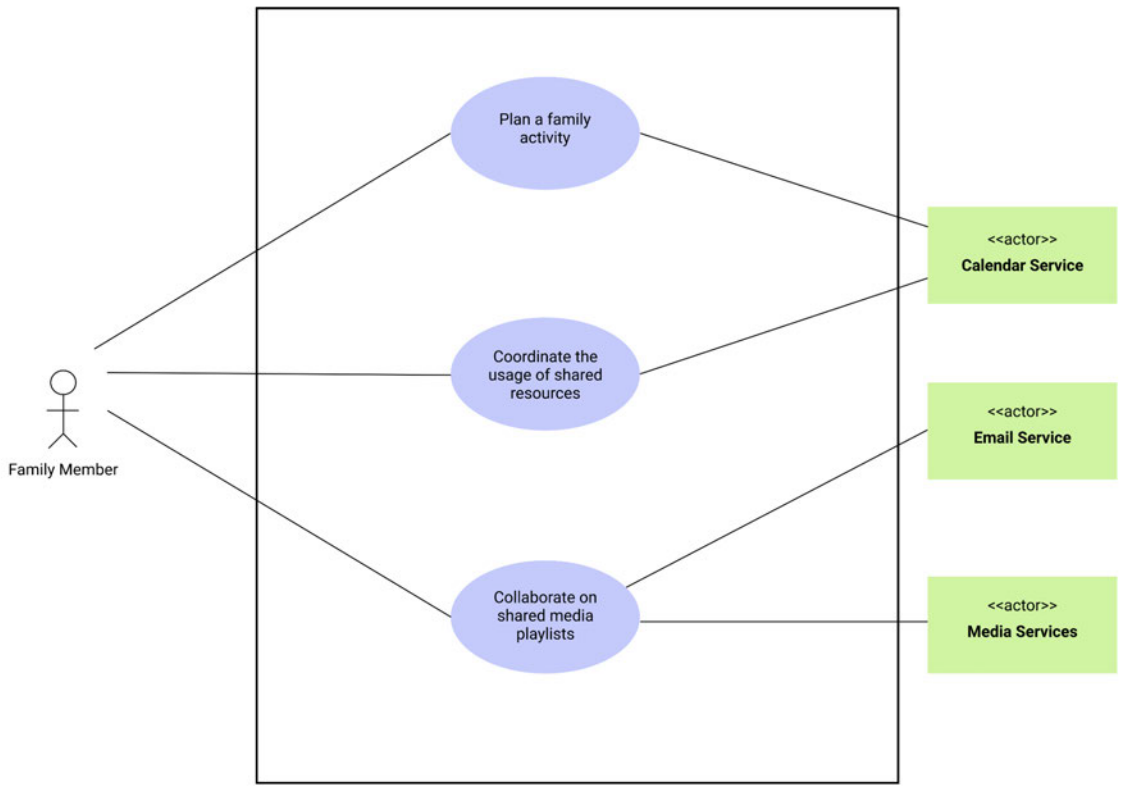

Figure 1: Use Case Diagram

Actor Descriptions

- **Family Member**: These are the primary human actors who are interacting with the system. Family members use the system to facilitate the planning of family activities (with respect to each member's individual schedule), coordinate how a shared resource is used among multiple family members, and collaborate with other family members on shared media playlists.

- **Calendar Service:** This is a non-human actor that provides our system with information about each family member's individual schedule, and the schedule of shared resources. Examples of family member's individual schedules include times when they are available and times when they are unavailable. An example of the schedule of shared resources is the time slots that shared resources are free to be booked out by a family member. Additionally, our system updates an individual's schedule on the calendar service with scheduled family activities.

- **Email Service:** This is a non-human actor that sends data, in the form of emails, to the family members' email accounts when a playlist is created, and the family member is invited to collaborate on it.

- **Media Services:** These are non-human actors that provide online media resources, which include videos and music, that can provide data to the playlists created on our system.

Use Case Descriptions

- **Plan a family activity:** The system allows family members to initiate, schedule, and host a family activity. This includes giving family members the ability to propose an activity for the whole family to participate in, vote on a specific activity, and check each participant's availability through the integration with the existing calendar service.

- **Coordinate the usage of shared resources**: The system allows family members to book time for when they want to use a certain shared resource and notifies all other family members of that usage. It will also allow family members to set priorities for who gets to use a resource at what time slot, and allow family members to input a reason and a priority level for booking out the resource in case conflicts arise.

- **Collaborate on shared media playlists:** The system enables family members to create shared media playlists consisting of different media types, including music, videos, and pictures for different occasions, such as family dinners, camping trips, and driving, through the integration with media services. This feature helps family members share common interests through media.

2.2 Product Features

This next section goes into the details of the features for each of the three use cases described in Section 2.1. The artifacts shown here include workflow models and user stories.

Workflow Models

The workflow models presented are activity diagrams that depict the general flow of control of the three use cases in our system. Following the UML convention for these activity diagrams, we are able to graphically represent the steps performed by a family member, the system and other actors in each of the use cases on the system. These activity diagrams show what steps are followed for the successful completion of each use case's workflow, but also include points of deviation from the happy path through decision points and failure termination states.

Here are the activity diagrams for each use case:

Use Case 1: Plan a Family Activity

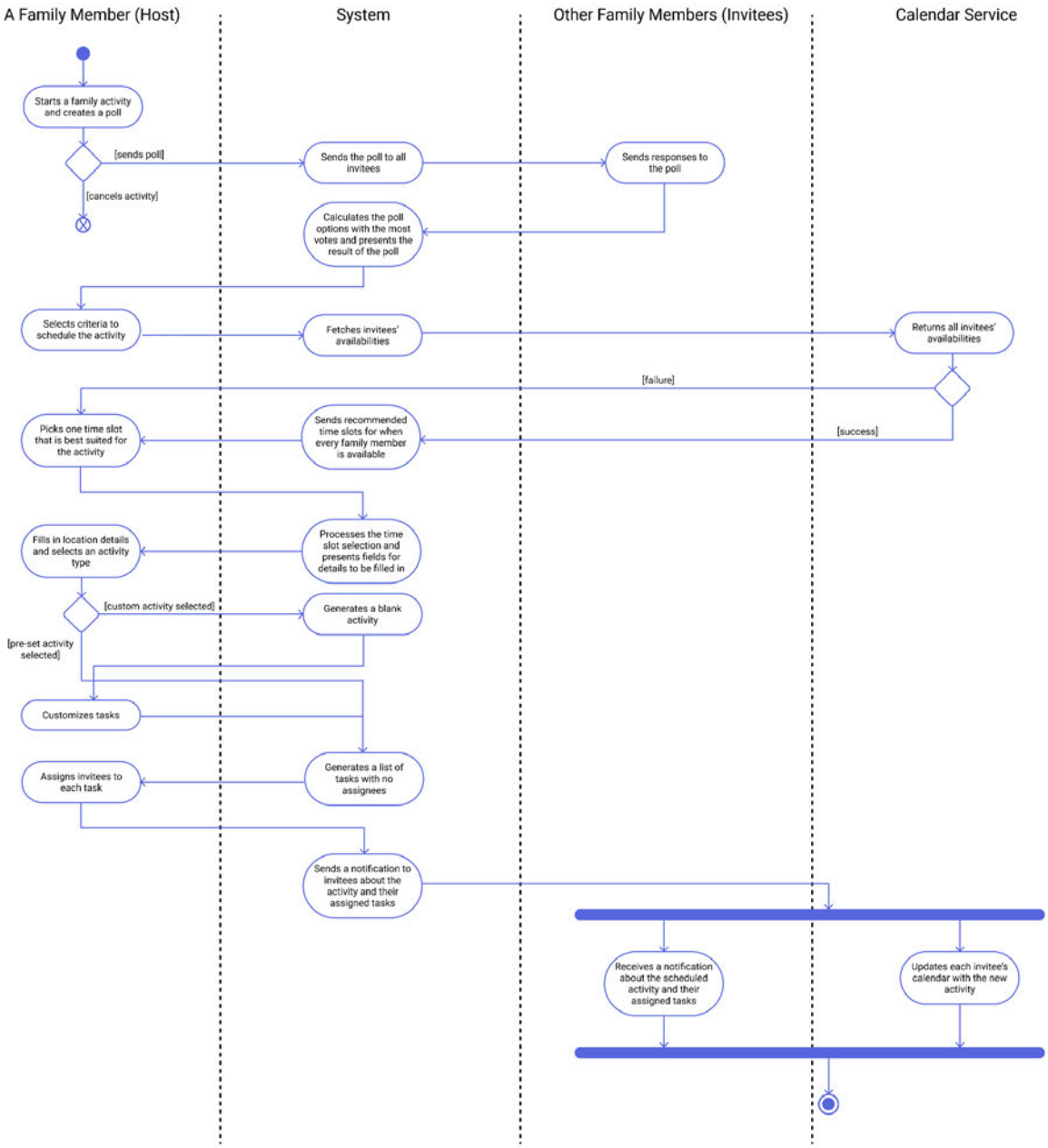

Figure 2: Activity Diagram for Planning a Family Activity

Description

In this model, the host starts by creating an activity and sending out a poll to the invited family members to get an understanding of which activities will be the most popular among the family. The system takes care of calculating which poll option is the most popular among the invitees who voted, and sends that to the host. Then, based on the host's specified scheduling criteria, the system checks each participating family member's calendar from an integration with the calendar service, and presents time slots that every invited family member (including the host) is available for. The host selects one of the time slots presented as the scheduled date and time for the family activity. Next, the host fills in the location that the family activity will take place at, and can select or create a custom family activity type. Then, using that family activity type, the system generates a list of tasks that can be assigned to other family members for the tasks that are required to be completed for the family activity. Lastly, the system finalises the activity planning by sending a notification to all family members about the activity and their assigned tasks. At the same time as the notification is being sent out, the system also updates the calendars of all family members who are participating in this new family activity through the integration with the calendar service.

Use Case 2: Coordinate the Usage of Shared Resources

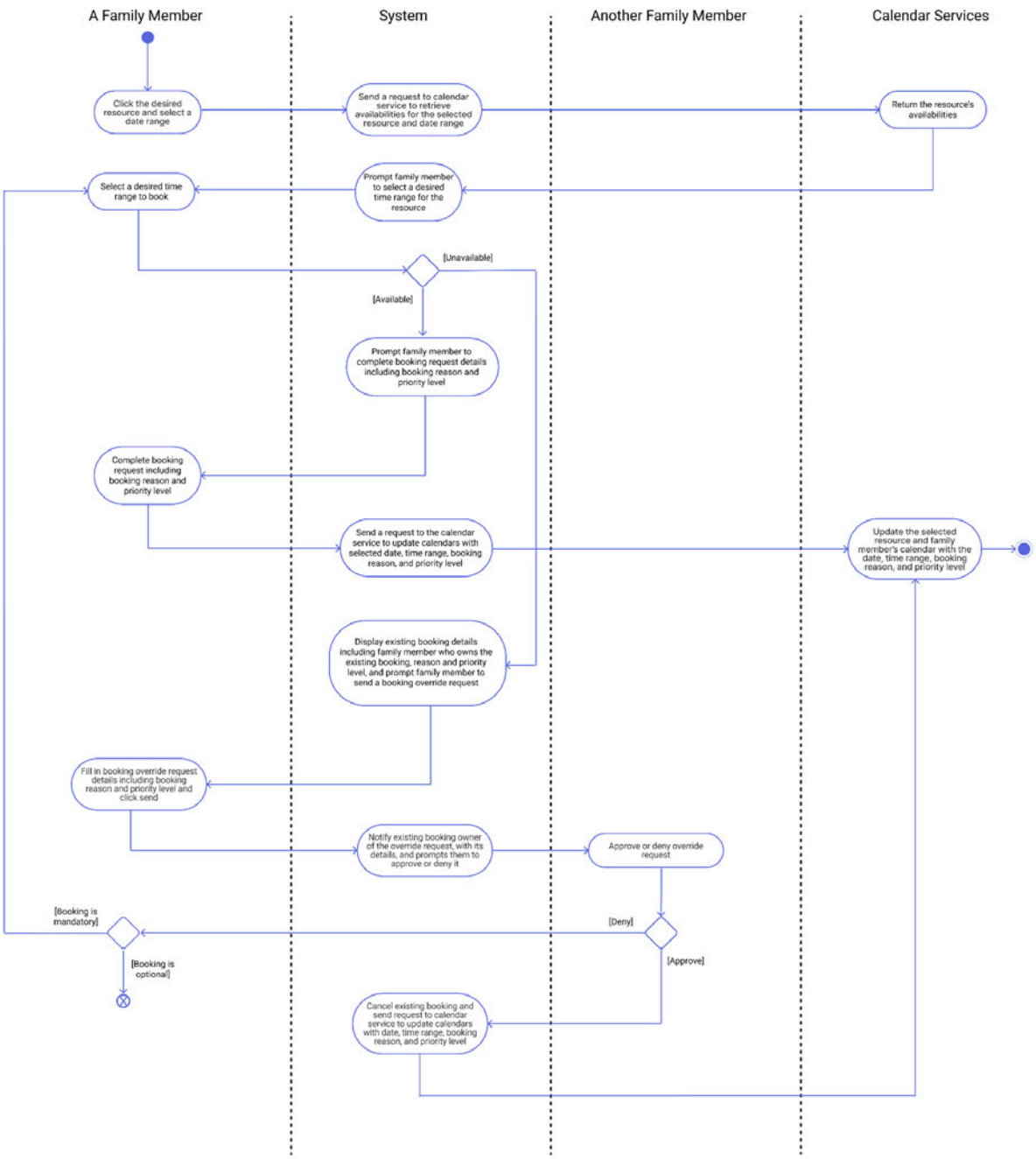

Figure 3: Activity Diagram for Coordinate the Usage of Shared Resources

Description

In this model, a family member who wants to use a shared resource among the family starts by selecting the desired resource and date range they want to book out. The system then retrieves the availabilities of the selected resource during the selected date range by sending a request to the calendar service, which returns the availabilities. The system then prompts the family member to select a time range within the selected date range to book, and the system will check if that time range is available by comparing it to the availabilities retrieved from the calendar service. If it is available, the system prompts the family member to complete the booking by entering booking details, including the booking reason and priority level. Once the family member completes these details, the system sends a request to the calendar service to update the resource and family member's calendars with the booking date, time range, reason, and priority level. Otherwise, if the selected time range is not available, the system prompts the family member to complete a booking override request, which also requires the input of the booking reason and priority level. Once the family member enters these details, the system notifies the existing booking's owner (another family member), to approve or deny this request. If the other family member approves it, the system cancels the existing booking and replaces it with the new one by sending a request to the calendar service to update the necessary calendars. If the other family member denies it, the system prompts the family member to re-enter a new time range and repeat the time booking process.

Use Case 3: Collaborate on Shared Media Playlists

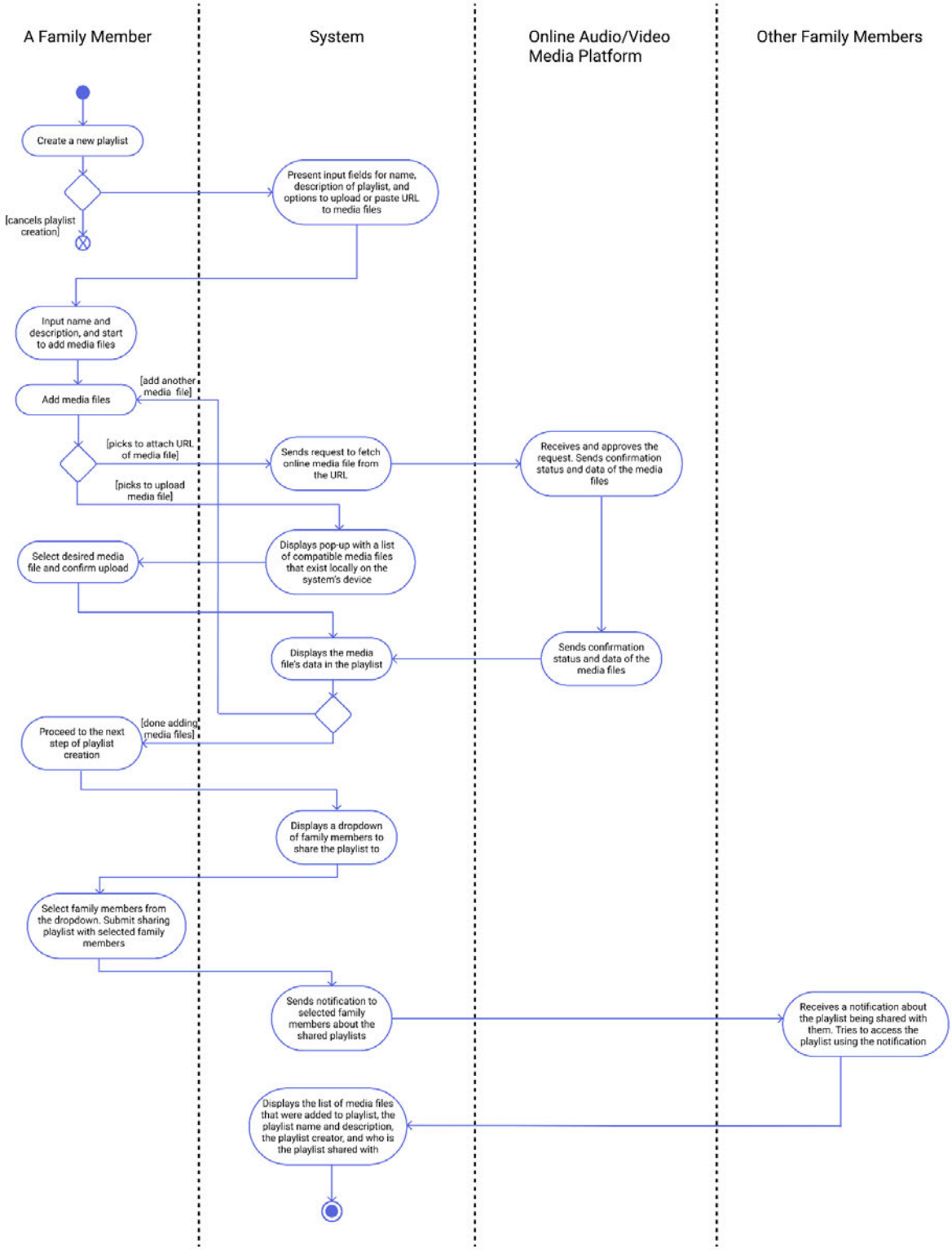

Figure 4: Activity Diagram for Collaborating on Shared Media Playlists

Description

In this model, a family member starts by creating a new playlist. The system requests input for the name and description of the playlist, and options to paste a URL of media files or upload them from their local device. After inputting the name and description of the playlist, the family member picks whether they want to upload a media file, or paste a link to a media file hosted on an online audio or video platform. If they choose to upload a media file, the system displays a pop-up window specifying all the compatible files that can be uploaded. If they choose to paste a URL, the system sends a request to the online audio or video platform that hosts the desired media file. After the host platform approves the request, it sends all the data related to the requested media file to the system, which then displays the information received on the playlist. After the family member adds all their desired media files to the playlist, they proceed to select family members they would like to share the playlist with. The family member who creates the playlist can select other family members from the dropdown of family members provided by the system, or they can manually input the emails of the family member if they do not appear in the dropdown. The selected family members are notified of the playlist by the system's own notification system, or through email. After receiving the invite, the selected family members are able to view, play and edit the playlist.

User Stories

A convenient method of providing a light-weight approach to managing requirements early on in the product development lifecycle is through the use of user stories. These are short statements, written from the family member's point of view, to express how a feature or functionality solves a problem for them. We have written our user stories for each of the three use cases in our system using the following format:

<u>As a <user/role>, I want <something>, so that <benefit is achieved>.</u>

Here are the user stories for each use case:

Use Case 1: Plan a Family Activity

Host POV:

- As a family member who hosts an activity, I want to select which family members to invite to a family activity, so that only family members who should be invited receive information about the family activity.
- As a family member who hosts an activity, I want to allow other family members to vote on a family activity, among different choices, so that I know all family members' willingness to participate.
- As a family member who hosts an activity, I want to find a period of time where all invited family members are available, so that all invited family members can participate in the family activity.

- As a family member who hosts an activity, I want to fill in all relevant details for the family activity, so that the family activity contains all necessary information for participating family members.
- As a family member who hosts an activity, I want to assign responsibilities for planning the activity to invitees, so that the burden of responsibility is spread out across all family members participating in the activity.

Invitees POV:

- As a family member who is invited to an activity, I want to vote on what the activity should be, so that I can express my willingness to participate.
- As a family member who is invited to an activity, I want to see the family activity reflected in my calendar, so that I can conveniently access the details of the activity.

Use Case 2: Coordinate the Usage of Shared Resources

Family member who wants to use a resource POV:

1. As a family member who wants to use a shared family resource, I want to indicate the priority level of using the shared resource, so that I can resolve possible resource usage conflicts.
2. As a family member who wants to use family resources, I want to indicate when I am using a shared resource, so that other family members are aware of when a resource is being used.

Other family member who may already be using a resource POV:

3. As a family member who wants to use a shared family resource, I want to know other family members' current usage of shared resources, so that I can plan my usage around their lifestyles.

Use Case 3: Collaborate on Shared Media Playlists

Creator POV:

- As a family member, I want to share different types of media content, so that my family members and I can establish common conversation topics.

Editors POV:

- As a family who is invited to collaborate on a playlist, I want to be able to play and edit the content in the playlist, so that I can contribute in creating more common conversation topics with my family.

2.3 User Characteristics

The main users of Modern Family 2031 are members of the same family living in the same household, residing in North America.

- Family members are young adults and busy parents who share a living space.
- Busy parents have to balance their work, social, and family life. They want to spend time with their children and bond over mutual interests, but are unable to do so due to their busy schedules, and generational gaps.
- Young adults mostly want privacy and do not like their spaces to be invaded. They love to use their phones and often spend a lot of time on it. However, they must share major resources with all family members because they are not capable of getting their own resources yet.
- Family members have a basic understanding of mobile devices (examples including installing applications, creating an online account, etc.), but do not need to be very tech-savvy.
- Family members have an email that can be used to create an account on Modern Family 2031.

Amy Ng

Hamilton, ON

Gender: **Female**

Age: **45**

Income: **$45,000**

Occupation: **Swim coach, cashier, food delivery driver**

Education: **Bachelor's Degree**

Techical Competence: **Struggles with new apps**

Bio

Amy Ng is a first-generation immigrant who works mainly as a swim coach. She has divorced with her ex-husband 8 years ago and she currently lives with her two teenage daughters in a condo. The older one is 17 years old and the younger one has just turned 13. They adopted a domestic shorthair cat which also lives with the family.

Amy's older daughter will go to university soon and her tuition is a big financial burden to the family. Being a swim coach is seasonal, so Amy decided to work part-time as a cashier and as a driver for food delivery apps to make some extra money. She works about 10 hours per day including the weekends, leaving herself little time to catch up with her family about their lives.

Personality

ISTJ - Introverted, Sensing, Thinking, Judging

Goals

- Bond with her relatives in Hong Kong
- Find a way to set up family events easily
- Get updates from her daughters even when she is at work

Pains

Low level of savviness in technology, which leads to a slow learning curve for complex social media apps.

"I want to make sure my family is doing well without digging through their activity on social media."

Figure 5: Persona of a Busy Parent

Mark Young

North York, ON

Gender: **Male**

Age: **17**

Income: **$0**

Education: **Currently in high school**

Techical Competence: **His cell phone is with him 24/7**

Bio

Mark is attending grade 11 at a high school in North York, ON. He lives in a semi-detached house with his parents, and 2 of his 3 siblings. He has an identical twin and is the second youngest in the family. His oldest sister recently became engaged, and moved in with her fiance.

He turned 17 recently and just got his G2 license with his twin. The family only has 2 cars, which is shared among 5 people.

His parents used to both work full time in-office. However, when COVID hit, everyone started working/studying at home.

Personality

ESFP - Extraverted, Sensing, Feeling, Perceiving

Goals

- Use the car whenever
- Convince parents to get faster wifi
- Help out around the house...sometimes

Pains

Constantly have to argue with siblings or parent over things like who gets the bathroom in the morning first, who can take the cars out Friday night etc.

"I don't want to always have to argue with my siblings over small things like who gets to play with the Switch."

Figure 6: Persona of a Young Adult

2.4 General Constraints

The technological, legal, and functional constraints that the system must adhere to are listed as follows:

- Parallel operations
    - The system must be able to run in the background to asynchronously retrieve and send data to the calendar service even if the family member has switched to another application. This upholds the system's performance requirement.
- Audit functions
    - The system must include functions that can verify the integrity of stored data in order to maintain quality of the system's functionality for users. This upholds the system's performance requirement.
- Safety and security considerations
    - The system must be implemented such that an initiative is taken to encrypt data so that data breaches are prevented and privacy laws are complied with in order to protect user data such as passwords, calendars, personal identifiable information, and so on. This upholds the system's security requirement.
- Standards and laws
    - The system must adhere to accessibility standards according to the Accessibility for Ontarians with Disabilities Act (AODA) and the universal Web Content Accessibility Guidelines (WCAG). This upholds the system's usability requirement.
- Criticality of the application
    - Measures must be implemented to handle down times of any adjacent systems so that the family members can continue using the system without experiencing system failure. This upholds the system's performance and usability requirements.

2.5 Assumptions and Dependencies

In order to satisfy use case requirements, there are some assumptions the system must make about the external environment and adjacent systems.

The following assumptions are made by the system for the top three use cases of this mobile application as well as some general, overarching system assumptions.

General Assumptions

- Every family member has an email that can be used to register for an account
- Every family member with an account remembers their email and password for login
- Every family member has a mobile device and reliable internet access
- Family members have enabled notifications for the mobile application on their local devices' settings

- The calendar service API [1] sends the correct data values back to our system, and in the expected format, at all times (for the resource sharing and plan a family activity use cases)
- Each family member account holder understands the system language, which is English
- Every family member intends to use the system to facilitate family coordination

Use Case 1: Plan a Family Activity

- Every family member has a registered family member account on the system
- Each family member has a personal calendar with an adjacent system our system is integrated with and all family members are willing to share their personal calendars with other family members
- All family members' calendar schedules are up-to-date when the scheduling data is sent to our system
- The family members that are invited to a family activity, by the host, are capable of participating in said activity
- The host family member uses the poll to determine the most preferred family activity for the family members to participate in, if the host chooses to use the poll feature
- If choosing to use a poll, all the options that the host includes in the poll are appropriate activities for the invited family members to participate in
- Each vote registered by the polling tool was inputted by a family member that is invited to the family activity the poll was sent for
- If the host chooses to manually enter a family activity, without using a poll, then the activity they enter is one that all invitees are willing to participate in
- If the host chooses not to use the calendar service to schedule the family activity, then the date and time they manually enter for the family activity is assumed to be a time slot that all participating family members are available for
- The location the host entered for the family activity is the actual location that the family activity will take place at
- When assigning tasks to participating family members, the host is assigning a suitable and capable family member to each task
- Family members participating in a family activity adhere to their responsibilities with regards to the tasks that the host has assigned them
- All family members want to see the details of the family activity reflected on their personal calendars that have been integrated with the system

Use Case 2: Coordinate Usage of Shared Resources

- Every family member has a registered family member account on the system
- All family members use the resource booking system whenever they want to book a shared resource
- All family members adhere to the booking override request process when there are resources sharing conflicts that arise
- The family member that is sent a booking override request decides who gets to use the resource based on the priority level and the reason of its booking request
- Family members have defined what each priority level means among the family
- Family members honestly enter the priority level of their booking
- Each resource that can be shared within the family has been registered into the system in the resources inventory
- All family members adhere to the usage time slots of the booking and update the time in a timely manner if there is a change

Use Case 3: Collaborate on Shared Media Playlists

- The family member who creates the playlist is the same person who is adding other editors
- All media files added to playlists are ones that can be enjoyed by the family members
- Adjacent online media platforms will send the correct data, in the expected format of the pasted URL, to the playlist if the retrieval is successful

3 Interface Specification

This section contains details about the technical aspects of the requirements and specifications related to the Modern Family 2031 system. This includes UML models, such as the specification domain models and state machine models (in the form of navigation diagrams), for each of the three use cases. Other models and diagrams include scenarios for each use case, and UI sketches for various screens on the application. Lastly, the quality requirements for the overall system are detailed as well.

3.1 Domain Models and UI Sketches

This section contains the specification domain models and various UI sketches related to the top three use cases of the system: Plan a Family Activity, Coordinate the Usage of Shared Resources and Collaborate on Shared Media Playlists.

Domain Models

The specification domain model represents a graphical data dictionary of the vocabulary that is used to describe the requirements and specifications of relevant environment and interface phenomenon related to the system. These models are split up into the three use cases, and using UML conventions, the goal is to express relationships between various entities in the system. Descriptions of entities and associations are included with the models.

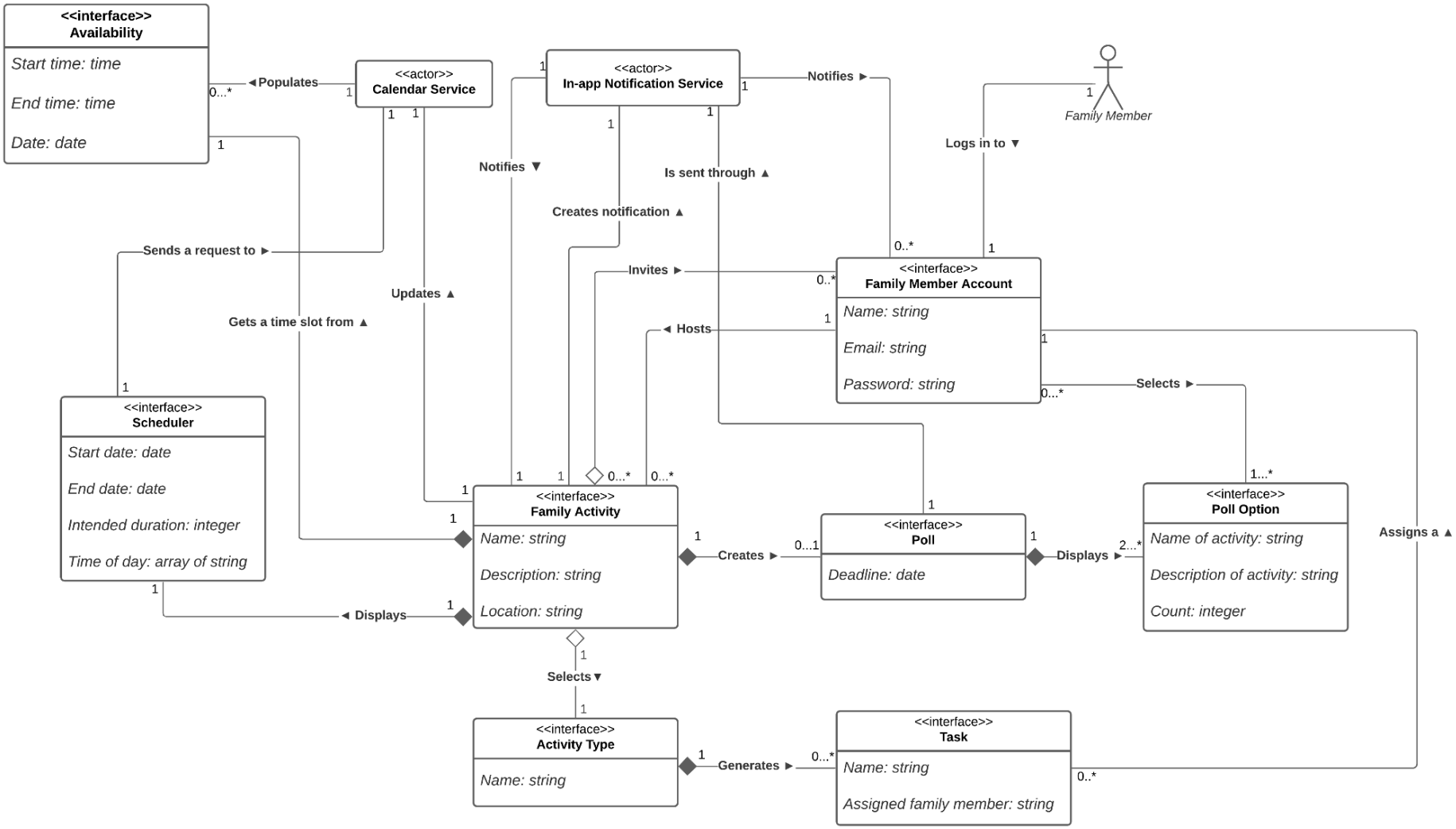

Figure 7: Domain Model of the Plan a Family Activity Use Case

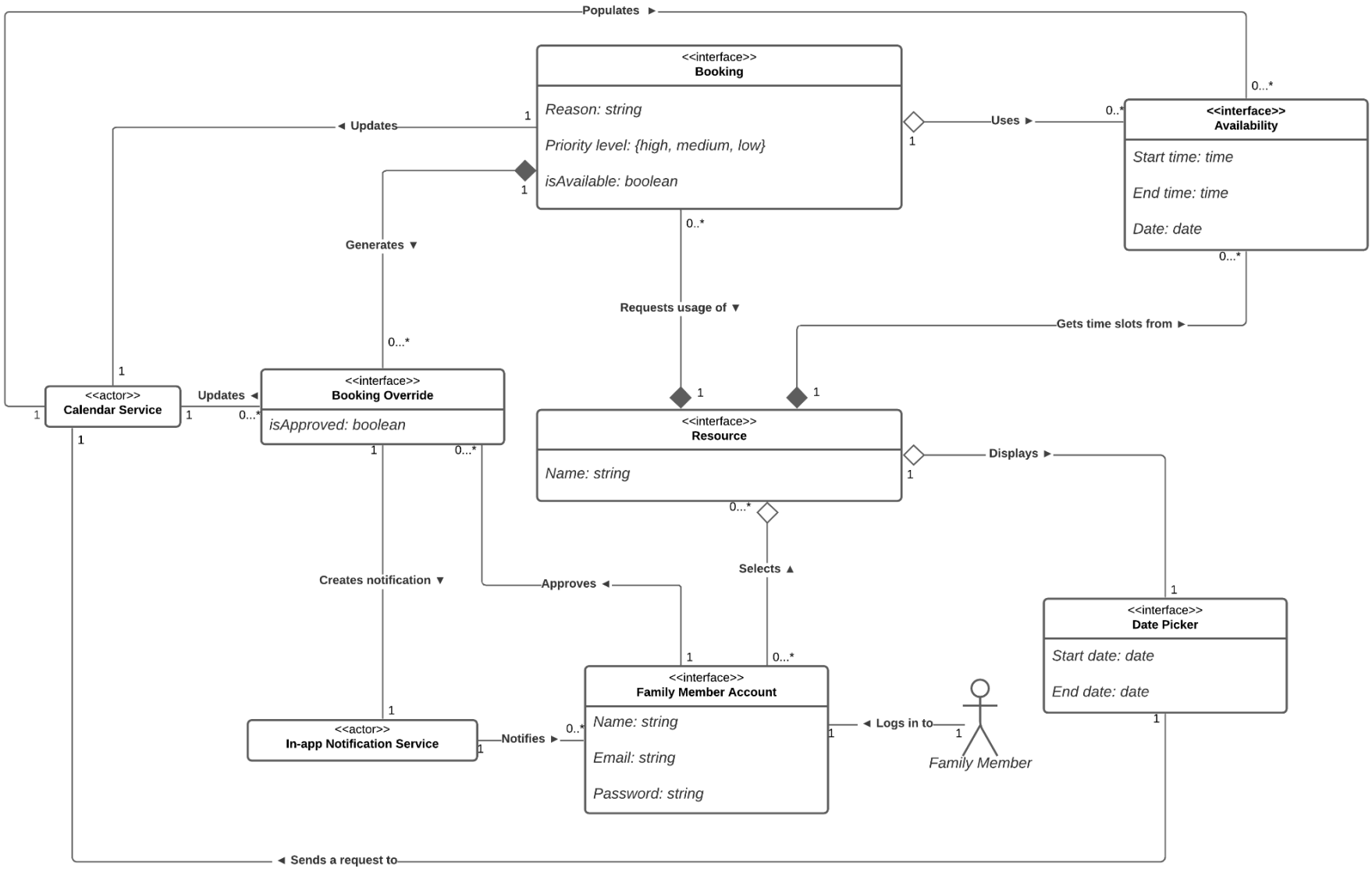

Figure 8: Domain Model of the Coordinate the Usage of Shared Resources Use Case

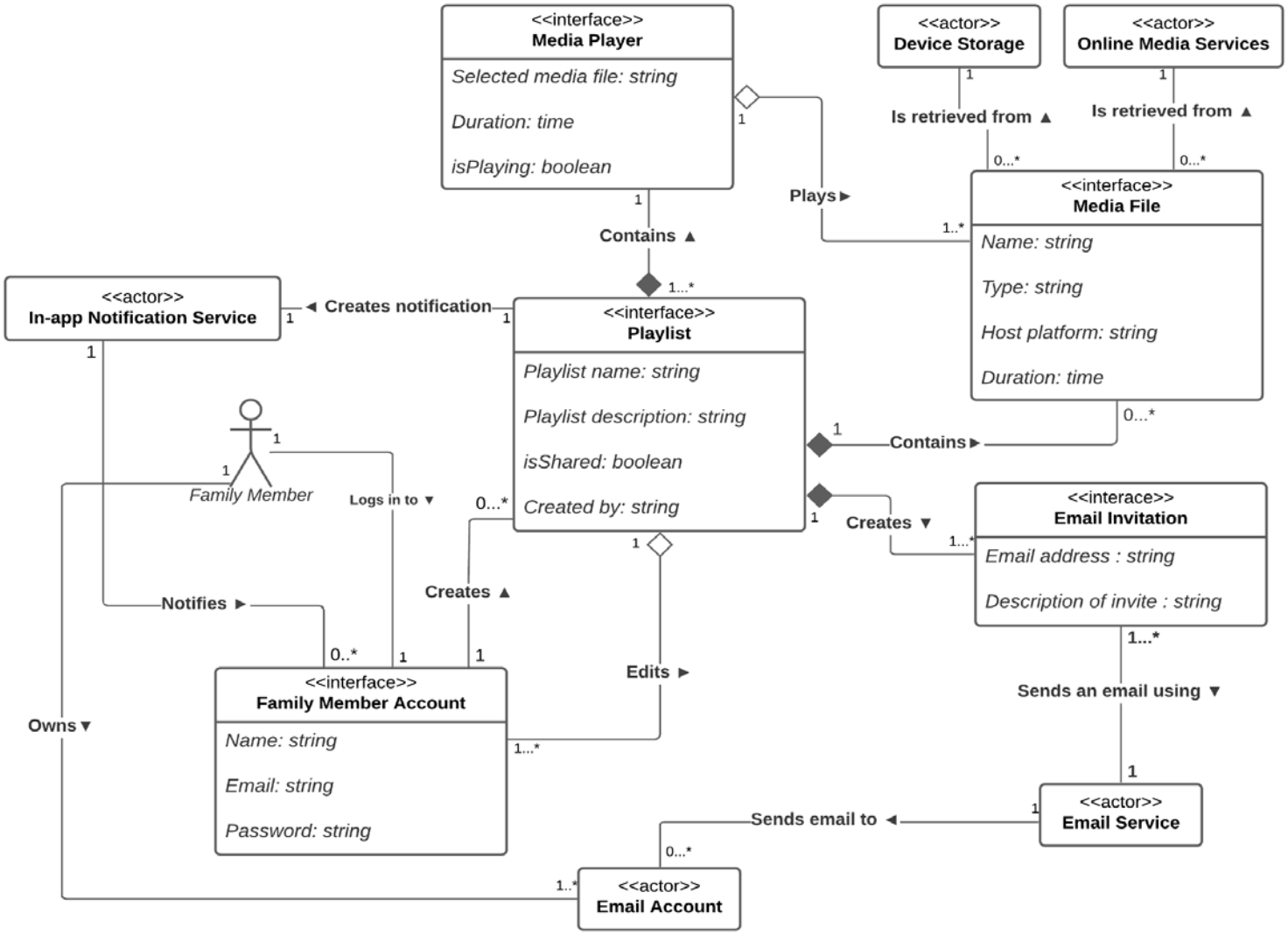

Figure 9: Domain Model of the Collaborate on Shared Media Playlists Use Case

General (Entity Descriptions)

Calendar Service
A service, provided by an adjacent system, that has data about the availability related to each family member account and each shared resource of the family.

In-app Notification Service
A service that constructs and displays corresponding messages depending on the given data.

Availability
An entity that shows an available time slot that can be booked for a resource or a family activity.

Family Member
A human actor who uses the system for family coordination.

Family Member Account
An entity that stores family member information when a family member signs up for the application, providing their name, email address, and password. The account enables family members to go through various workflows on the system.

Use Case 1: Plan a Family Activity (Entity Descriptions)

Family Activity
An interface that allows the family member accounts to host a family activity by scheduling a date and time, inviting other family member accounts, adding a location, and optionally creating a poll to vote for a family activity that the majority of the invitees prefer to attend.

Poll
An interface that includes poll options that each invitee can vote for, with regards to the options the host inputs for potential family activities. It also has a corresponding deadline that specifies the latest time for voting, and the poll is closed when the deadline passes.

Poll Option
An option that is subject to a poll, which contains the name and the description of a family activity, so family member accounts can view the details of a proposed family activity. It also includes the count of the votes for each option, so the system can see which option is the most popular among votes submitted by the invited family members.

Scheduler
An interface that allows the host family member to specify scheduling criteria, including a start date, end date, intended duration of a family activity and the time(s) of day that the activity would occur during. The information in this entity is used to fetch available time slots for all invited family members from the calendar service.

Activity Type
An interface that allows the family member to select a preset type for a family activity or create a custom one.

Task
A responsibility, based on the family activity type, which can be assigned to an individual family member account.

Use Case 2: Coordinate the Usage of Shared Resources (Entity Descriptions)

Booking
An interface to input the reason and the priority level of using a resource. If the usage time range is not within the fetched availabilities, the booking notes the selected time range as unavailable through the attribute isAvailable.

Booking Override
A request that is used when multiple family members want to use the same resource during overlapping time ranges.

Resource
A representation of a shared family resource, such as a car, a monitor, or a house key. The interface simply contains the name of the resource.

Date Picker
An interface for the family member to select a desired date range which narrows down the data returned by the calendar service.

Use Case 3: Collaborate on Shared Media Playlists (Entity Descriptions)

Online Media Services
An adjacent system that serves as a digital media service (e.g. Spotify, Apple Music, Youtube, etc.) that provides data in the form of media files.

Device Storage
Device storage that contains media files, on a family member's local device, that the family member may want to upload to the playlists.

Media File
An entity that represents media, such as a song, video, picture, etc. It stores the name, the type, the host platform, and the duration of the media file.

Playlist
An interface to create, view, play, and edit a list of media files that family member accounts have added. It supports sharing as well, which enables other family member accounts, who are not the creator, to edit the playlist.

Media Player
An interface to play a media file. All playlists share one media player, and it stores the current media file that has been selected to play.

Email Invitation
An interface to allow the family member to preview an email invitation, and enter the email addresses of family members to share the playlist with.

Email Service
A service that sends data, in the form of email invitations, to the family member accounts when a playlist is created, and a family member account is invited to collaborate on it.

Email Account
An email account is external to the system and is a service that allows family members to send and receive emails. It is assumed every family member owns an email account.

General (Relationship Descriptions)

A Family Member logs in to a Family Member Account
The human actor, which is the family member, is associated with an account in our system. Since the system is only aware of the family member's account, it acts as the representative of the human actor in our modelling. This relationship is one-to-one as there is one account per family member.

The In-App Notification Service notifies the Family Member Account
Our in-app notification service receives a trigger from various other entities in our system and fires the alert to the appropriate family member accounts. These triggers are referenced in all three of our use cases. This relationship is one-to-many since the service can fire alerts to any number of family member accounts.

Use Case 1: Plan a Family Activity (Relationship Descriptions)

A Family Member Account hosts a Family Activity
The family member account that would like to initiate a family activity can choose to host one. This substantiates an instance of a family activity. This relationship is one-to-many as a single account can host as many activities as they please.

A Family Activity invites a Family Member Account
Once the family activity has been finalized, other family member accounts are invited to partake in the activity. Multiple family activities can send invites to multiple family member accounts, hence the relationship is many-to-many. Moreover, many family activities are experienced by one or more family member accounts, hence the relationship is aggregation.

A Family Activity creates a Poll
Once the family member account, that hosts the family activity, creates an instance of the family activity, a poll can be created. A family activity can have 0 or 1 polls created for said activity. Furthermore, this is a composition relationship because each poll is created from an instance of a family activity.

A Poll displays Poll Options
The poll can display multiple poll options. Since a single poll can contain 2 or more poll options (as it would not make sense to create a poll with less than 2 options), this is indicated by the

multiplicity symbols. If a poll is deleted, then its poll options are subsequently deleted, hence the composition relationship.

A Poll is sent through the In-app Notification Service
The family activity poll must be sent to all invitees of the family activity. This is done through the in-app notification service. Because there is only one family activity poll being sent to the notification service, this is a [1, 1] relationship.

A Family Member Account selects Poll Option(s)
A family member, represented by their account, can select their preferred family activity. Each member is allowed to make any number of selections but must select at least one preferred activity from the poll, which is indicated by 0…* and 1…* multiplicity symbols.

A Family Activity selects an Activity Type
Each family activity has one associated activity type. This relationship is one-to-one, as denoted by the multiplicity symbols. Since each family activity can have one family activity type, this is marked by the aggregation relationship.

An Activity Type generates Task(s)
The family activity type listed under the family activity is used to generate 0 or more tasks as indicated by multiplicity symbols. If the family activity type no longer exists, then the task no longer exists as well. This is indicated with the composition relationship. These tasks are used to plan the activity and are delegated to family members.

A Task assigns a Family Member Account
Each task will have a family member account assigned to it by the host. Since any number of tasks can be assigned to one family member account, this shows the many-to-one relationship between these entities.

A Family Activity updates the Calendar Service
Once the family activity is finalized, each participating family member's calendar must be updated with the date, time and other associated details of the activity. A family activity can update multiple family member calendars, however, the service is only pinged once to bulk update these calendars. This is indicated by the one-to-one relationship.

A Family Activity displays a Scheduler
The family activity displays the scheduler interface where family members can specify the criteria for scheduling the activity. This includes selecting a range of desired dates to have the family activity occur on, the intended duration of the activity and the time(s) of day that the family activity should occur during.. Each family activity has only one scheduler interface as indicated by the one-to-one relationship between these entities. Furthermore, each family activity has its own scheduler interface. This is marked by the composition relationship.

A Scheduler sends a request to the Calendar Service
The scheduler narrows down the selection of data that is requested from the calendar service. Since a single scheduler sends a request, based on the date range, intended duration and time(s) of day to the calendar service, this shows the one-to-one relationship between the entities.

The Calendar Service populates an Availability
The calendar service can obtain the various dates and times in which all family members are available, based on the criteria from the scheduler. This populates the availability interface entity. Because the calendar service can populate multiple availabilities based on the scheduling criteria, there is a one-to-many relationship between the calendar service and the availability.

A Family Activity gets a time slot from an Availability
A family activity can only happen during one time slot that is denoted by an availability. Hence, a single availability will be selected for the family activity, thus showing the one-to-one relationship between these entities. As the availability belongs to the family activity, this is marked by the composition relationship.

A Family Activity creates a notification via the In-app Notification Service
Once the family activity has been finalised, a notification is triggered via our in-app notification service. The service then fires the alert to all the associated family member accounts that are invited to the family activity. Since one family activity creates one notification, this is shown through the one-to-one relationship.

The In-app Notification Service notifies a Family Activity
A notification is sent to the family activity once the poll deadline has passed and no family members voted for activities through the poll. In this case, the host family member will receive the notification and, from there, manually enter an activity name and description before proceeding to the scheduling step for the family activity. As one notification gets sent to one family activity, this marks the one-to-one relationship between these entities.

Use Case 2: Coordinate the Usage of Shared Resources (Relationship Descriptions)

A Family Member Account selects a Resource
The family member account can select a resource from the family's inventory of resources. They may wish to select multiple resources to use, and each resource can be selected by multiple family member accounts, so this is a many-to-many relationship. Moreover, many resources are shared by one or more family members, hence the relationship is aggregation.

A Resource gets a time slot from an Availability
A given resource is available only during certain time slots. Hence, it can get a time slot from any of the availabilities. Because a resource can have multiple availabilities, this is denoted by one-to-many multiplicity symbols. Also, if the resource no longer exists, then its associated availabilities are deleted. This is portrayed through the composition relationship.

A Booking requests usage of a Resource
A given resource also has a booking interface. Because a resource can have multiple bookings associated with it, this is marked by one-to-many multiplicity symbols. If the resource no longer exists, its associated bookings are deleted which is indicated by the composition relationship.

A Resource displays a Date Picker
A given resource requires a date picker interface, such that the family member may input the date range for which they would like to book the resource. Furthermore, multiple resources can have the same date picker interface. This is marked by the aggregation relationship.

A Booking updates the Calendar Service
Once a booking is finalised, the resource's calendar must be updated via the calendar service. Since a booking pings only one calendar service, this relationship is one-to-one.

A Booking uses an Availability
A resource booking uses time slots from an availability. Once the booking has been deleted, it frees up its associated availabilities for the resource, so it is an aggregation relationship. Furthermore, for one booking, there may be 0 to many availabilities, hence the one-to-many relationship.

A Booking generates a Booking Override
If a family member would like to override another family member's booking, then a booking override interface is required. Each booking generates a booking override interface. This relationship is one-to-many as there could be multiple overrides per booking. Moreover, once a booking is removed, its booking overrides are also deleted as marked by composition.

A Booking Override Request creates notification via the In-app Notification Service
Once a booking override request is created, a notification is triggered via our in-app notification service to let the original booking owner know that they need to approve the request. This is a one-to-one relationship.

A Family Member Account approves a Booking Override Request
The family member of the original booking can now approve or deny a booking override request. This relationship is one-to-many as there can be multiple override requests that a family member account can approve or deny.

A Booking Override requests updates the Calendar Service
If a booking override has been approved, then it must request the calendar service to update a family member's calendar by cancelling the original booking and updating it with the new one. This relationship is many-to-one as there could be multiple overrides requesting to update via the calendar service.

The Calendar Service populates an Availability
The calendar service can obtain the various dates and times in which the resource is available, based on the date range stored in the date picker. This populates the availability interface entity. Because the calendar service can populate multiple availabilities based on a single date range selection, there is a one-to-many relationship between the calendar service and the availability.

A Date Picker sends a request to Calendar Service
The date picker narrows down the selection of data that is requested from the calendar service. Since a single date picker sends a request, based on the date range, to the calendar service, this shows the one-to-one relationship between the entities.

Use Case 3: Collaborate on Shared Media Playlists (Relationship Descriptions)

A Family Member owns an Email Account
Each family member owns an email account, which is used to send and receive emails. We assumed that every family member has an email that can be used to register for an account. A family member can own multiple email accounts, which is denoted by the 1 to [1…*] relationship.

A Family Member Account creates a Playlist
A family member, represented by their account, can create a new media playlist. Each account can create multiple playlists as denoted by one-to-many multiplicity symbols.

A Family Member Account edits a Playlist
A family member, represented by their account, can edit a media playlist. Moreover, a playlist can be edited by 1 to many family members. The playlist contains many family member accounts that can edit it as represented by the aggregation relationship.

A Playlist creates an Email Invitation
Once a playlist is created, the creator may wish to share the playlist with others. Doing so creates an email invitation entity. This is a one-to-many relationship because a playlist can create as many email invitations as needed. This is also a composition because email invitation is owned by a playlist.

An Email Invitation sends an email using the Email Service
Once the email invitation is created, the email service is pinged to send out the email invitations to collaborate. This is a many-to-one relationship because multiple email invitations can request the email service at the same time.

The Email Service sends an email to an Email Account
The email service completes the issuing of email invitations to an email account, which is owned by a family member. This is a one-to-many relationship as the email service can send the same email invite to multiple email accounts.

A Playlist contains a Media Player
To play the media files within a playlist, it must contain a media player entity. Since there are potentially many playlists and only one media player, this is a [1, 1..*] relationship. Moreover, since a media player cannot be accessed if there are no playlists, this is a composition relationship.

A Media Player plays Media File(s)
The media player entity plays the individual media file(s) found within a playlist. Since there is only one media player for one or more media file(s), this is marked by [1, 1…*] relationship.

A Playlist contains a Media File
A playlist is made up of multiple media files. This relationship is one-to-many as a single playlist typically contains multiple media files. Moreover, if the playlist is deleted then, in the context of our system, the media file is also deleted. Hence, this is denoted by the composition relationship.

A Media File is retrieved from the Device Storage
A media file can be retrieved from local device storage, such as from a family member's phone. The file is then displayed in a playlist. This relationship is many-to-one as multiple media files can be obtained from device storage.

A Media File is retrieved from Online Media Services
A media file can also be retrieved from an online media service (e.g., the cloud). The file is then displayed in a playlist. This relationship is many-to-one as multiple media files can be obtained from the online media service.

A Playlist creates a notification via the In-app Notification Service
Once the media playlist has been created and shared with family members, a notification about how a playlist is shared with them is triggered via our in-app notification service. The service then fires the alert. This is a one-to-one relationship between the playlist and the in-app notification service.

UI Sketches

The UI sketches represent light-weight mockups of what the interface will resemble for various screens on the application. The focus is on the content of each screen; the scope of what to include on a screen, and how to lay it out. These UI sketches are also split up into the three use cases and include annotations for the components of each screen.

Below is a summary table of the eleven UI screens:

No.	Screen Name	Description
1	Initialization Screen #1: Login screen	This screen provides an interface for the family member to log in with email and password credentials, or modify the password if they already have an account. Otherwise, the family member can create a new account.
2	Initialization Screen #2: Family centre menu page	This screen provides a full menu for family members to navigate to various areas of the app such as resource booking, media playlist creation and more.
3	Family Activity Screen #1: Host family member creates a poll	This screen provides an interface for the family member, who hosts the family activity, to create a poll of family activity options to send out for the invitees to vote on.
4	Family Activity Screen #2: Other family members vote on a poll	This screen provides an interface for family members, who are invited to the family activity, to vote on their preferred family activity options.
5	Family Activity Screen #3: Host family member schedules a family activity	This screen provides an interface for the host family member to select a desired date range, intended duration and time(s) of day for the family activity to occur on. It also allows the host to select a time from the available time slots generated by data returned from the calendar service.
6	Family Activity Screen #4: Host family member fills out other activity details (location, activity type and assigning tasks)	This screen provides an interface for the host family member to add the location and select the activity type for the family activity. Based on the selected activity type, tasks will appear and the host can assign these tasks to various family members.
7	Resources Screen #1: Family member selects resource and date range	This screen provides a dropdown for family members to select what resource they would like to book as well as a date picker for them to choose a date range when they want to use the resource.
8	Resources Screen #2a: Family member fills in booking form	This screen provides textual information about the selected resource, a time picker for family members to choose the time slot that they want to use the resource during, a single select option for family members to choose a priority level, and a text field for them to enter a reason for their booking.
9	Resources Screen #2b: Family member fills in booking override request	This screen is a variation of Resources Screen #2, where the time slot the family member chooses is unavailable. In this case, the screen displays a message showing who holds the current booking and prompts the family member to send a booking override request with priority level and reason for booking.

10	Media Playlist Screen #1: Playlist creation	This screen shows the first step, which is setting up basic info of the playlist and adding media files to it, in the creation of a shared media playlist. The second step in the playlist creation is to share it with family members. In the first step, the family member must enter the name of the playlist. They can also enter a description of the playlist if they wish. Then, they would proceed to add the media files from links for uploading it from their own device.
11	Media Playlist Screen #2: Playlist view	This is the view of a created playlist with the added media files. Family members who are invited to the file have the option to edit the playlist. They are able to see details of each media file such as name, length of the file, host of the file etc. They can also select a specific media file to play it.

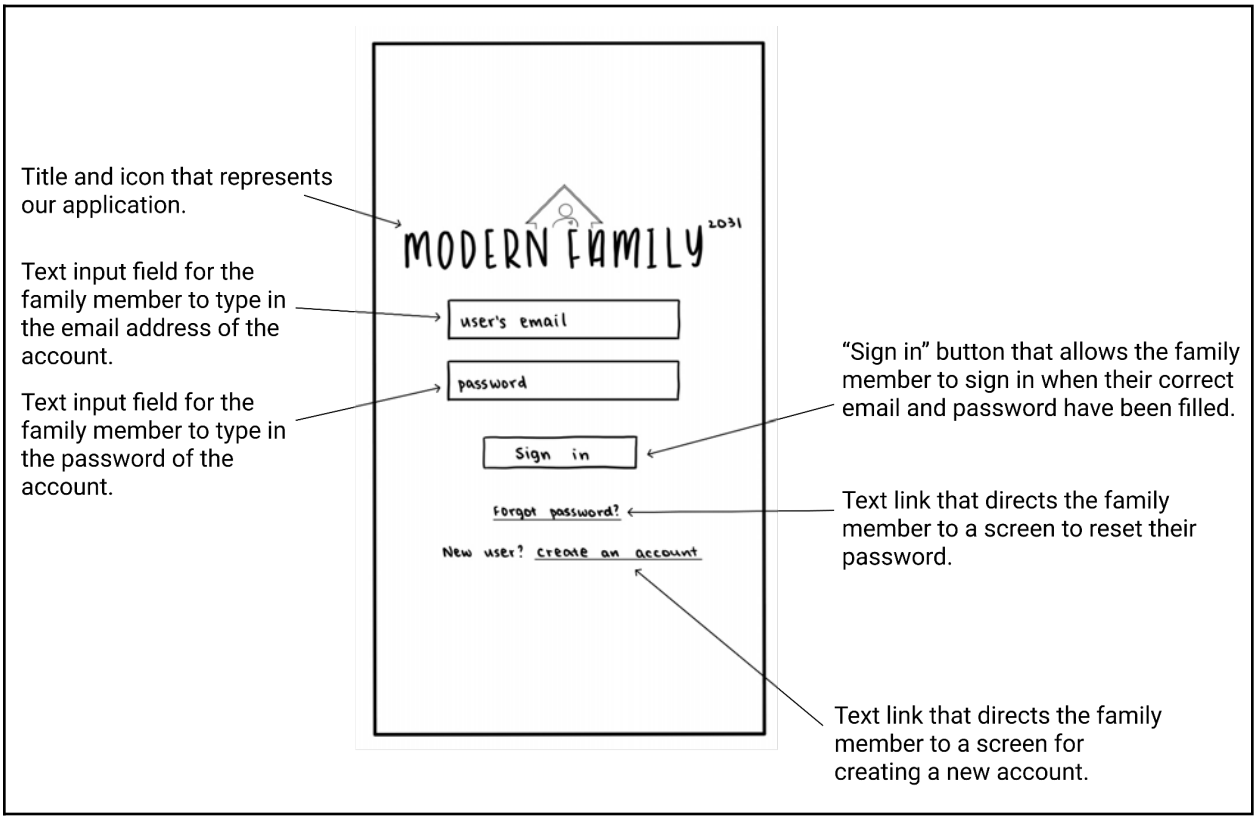

Figure 10: [1] Modern Family 2031 Login/Sign Up Screen

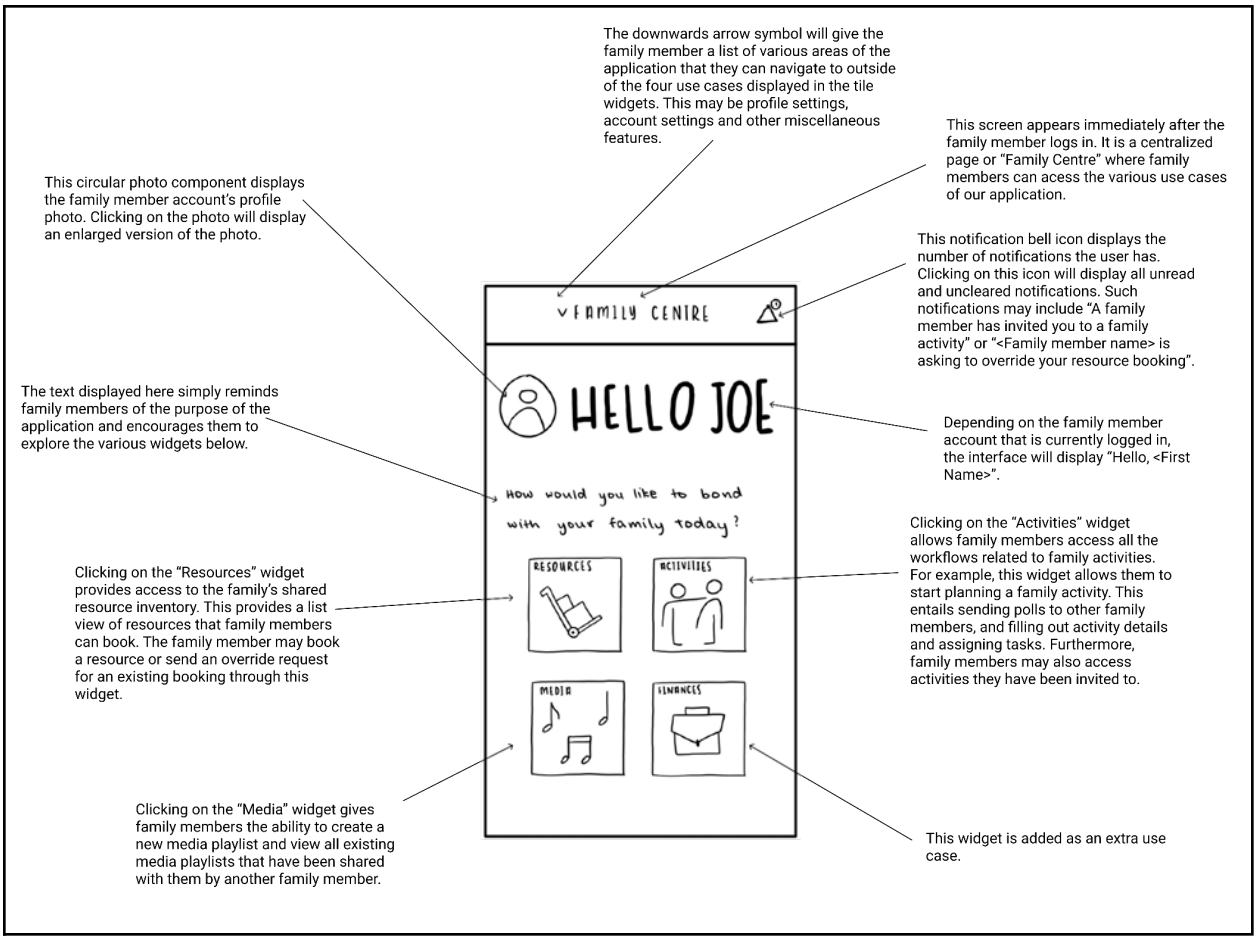

Figure 11: [2] Menu Screen to Explore Features

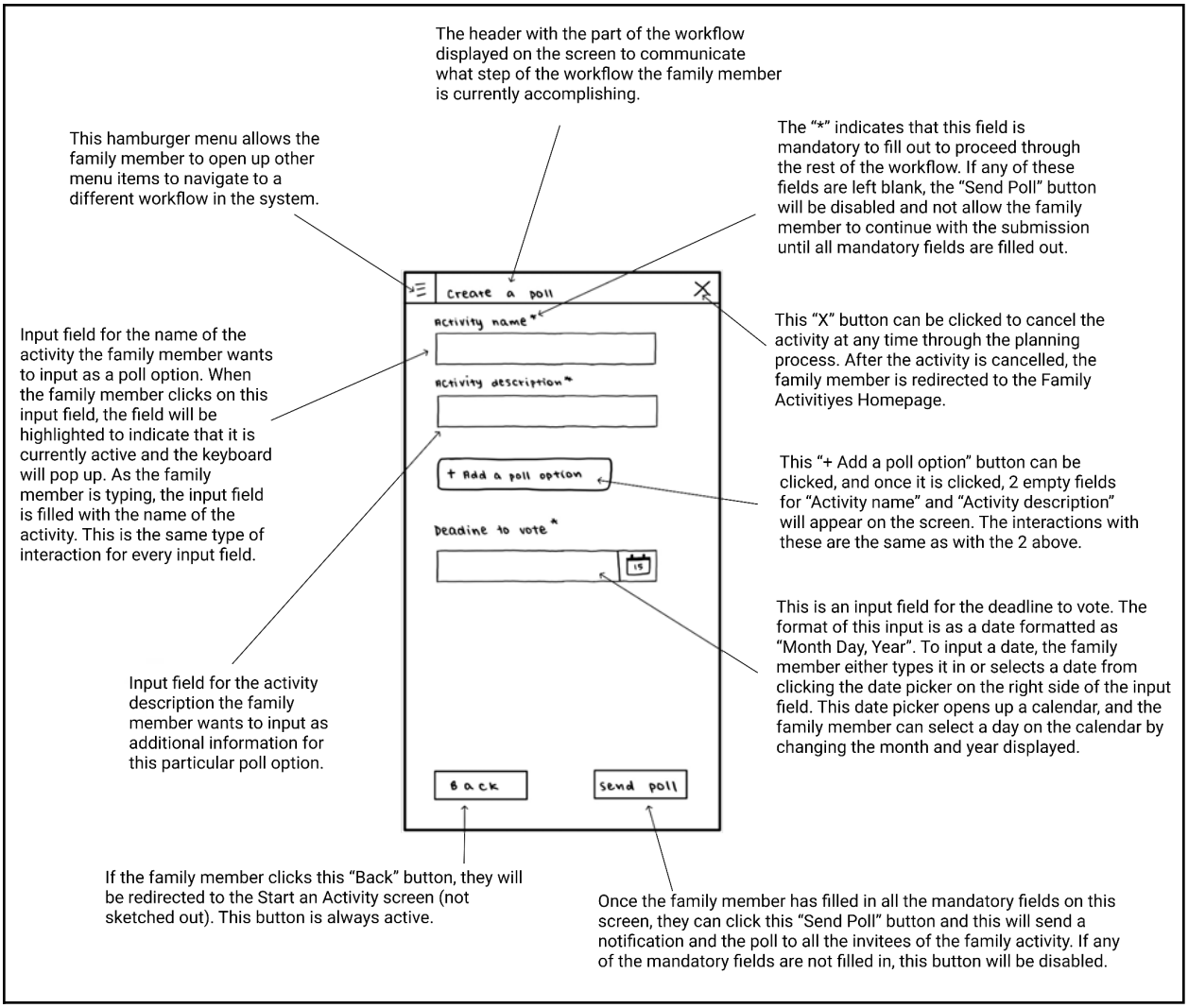

Figure 12: [3] Poll Creation Screen

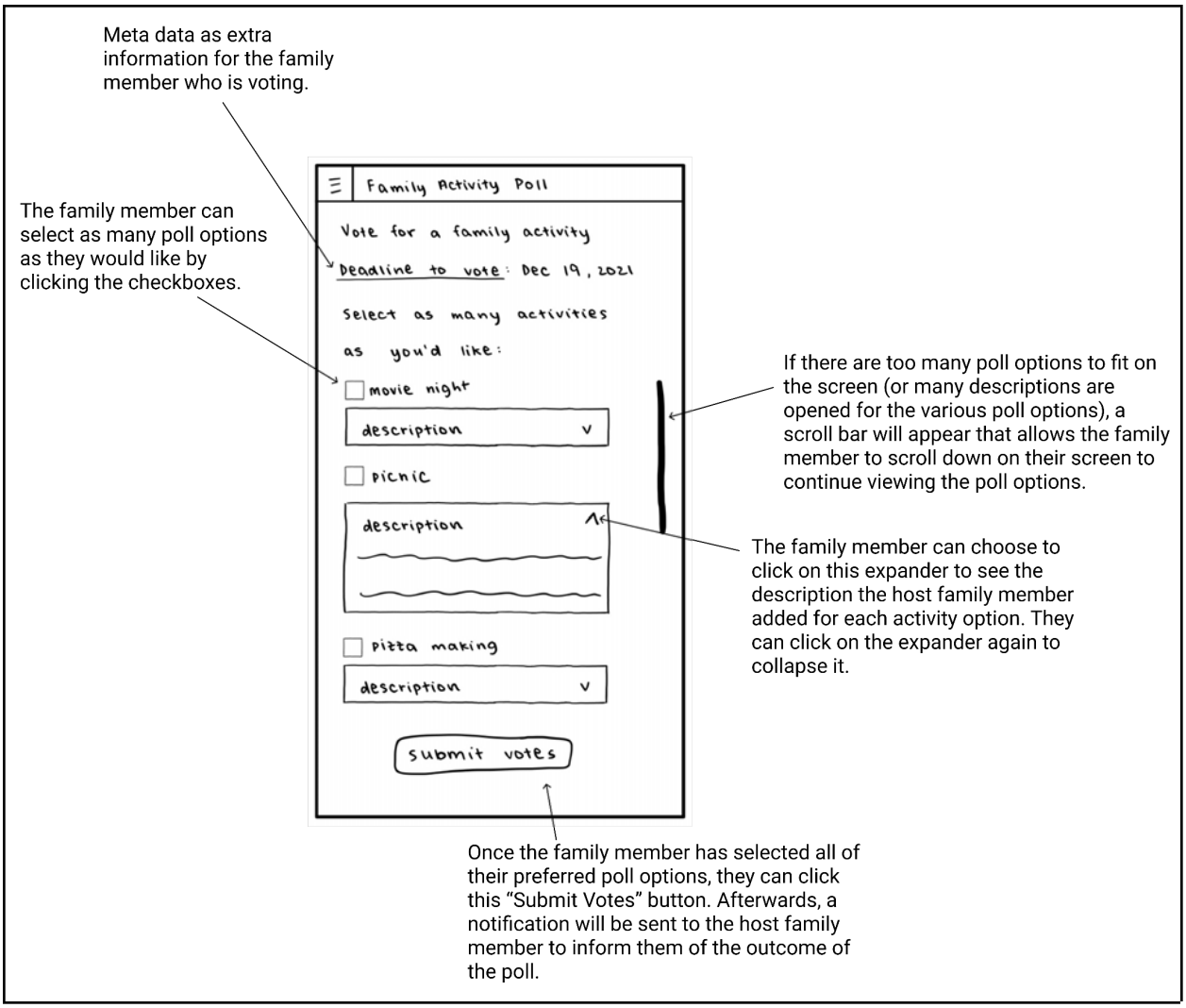

Figure 13: [4] Screen to Allow Invitees to Vote on Poll Options

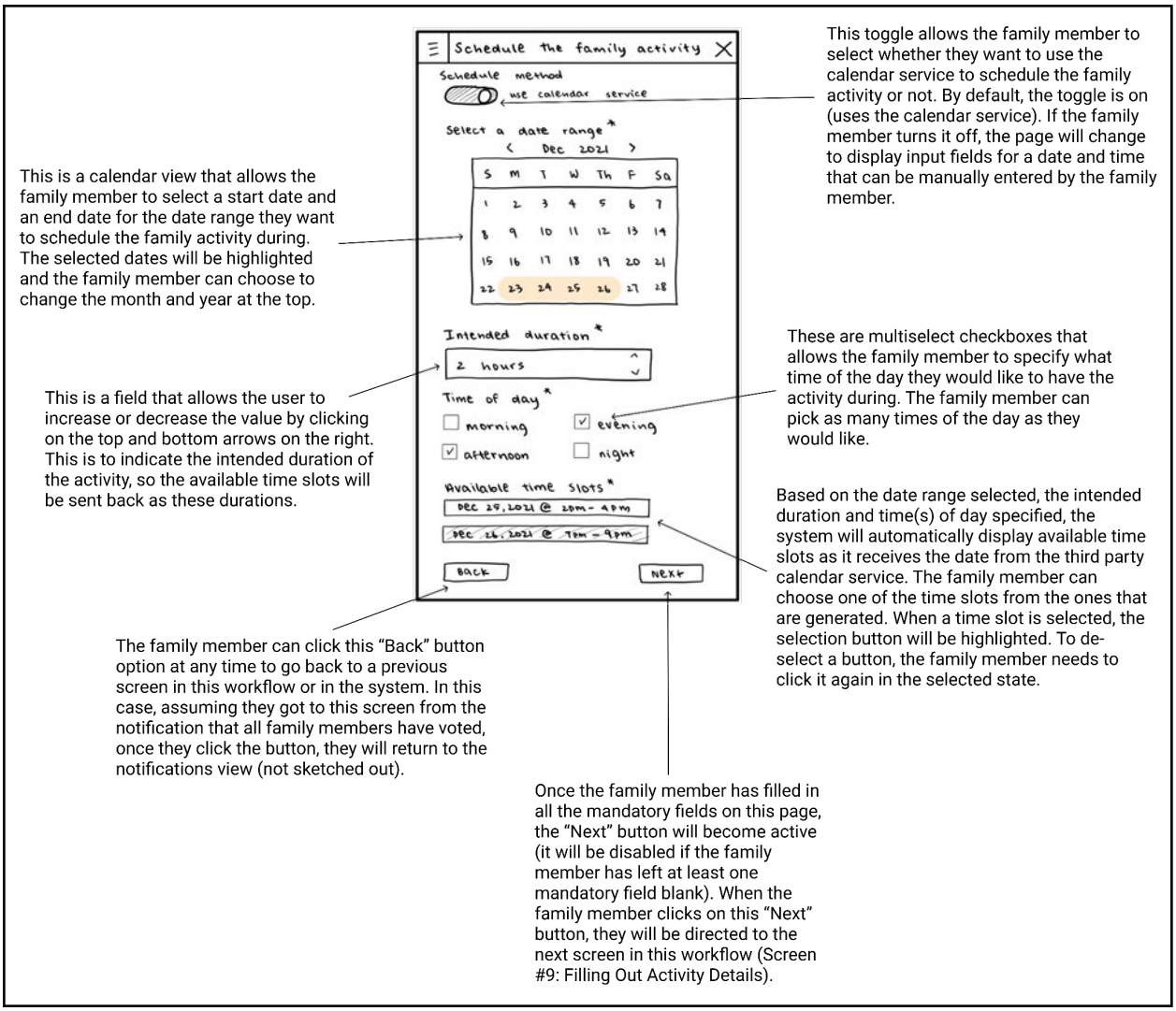

Figure 14: [5] Screen to Schedule the Activity

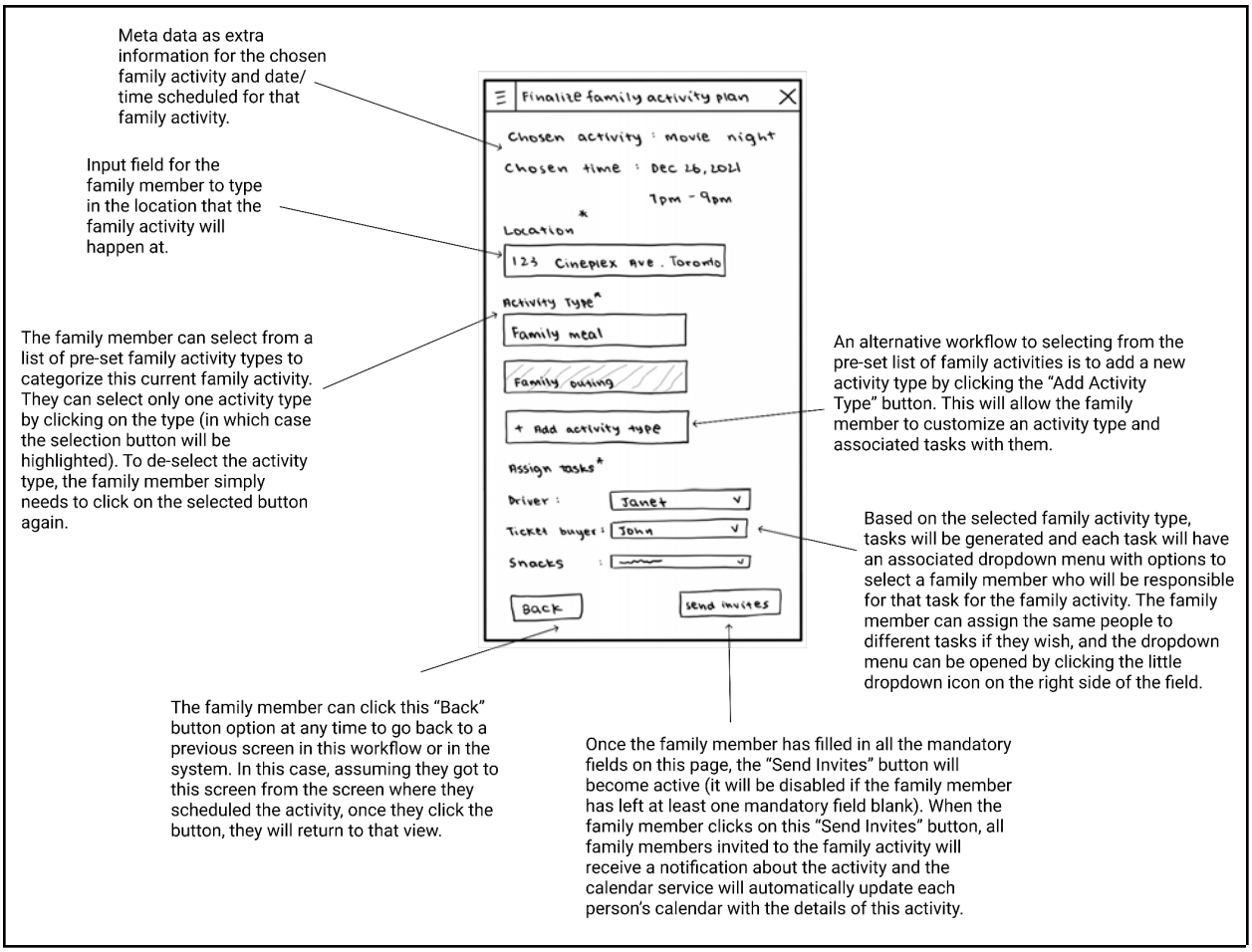

Figure 15: [6] Screen to Fill In the Details and Assign Tasks of the Activity

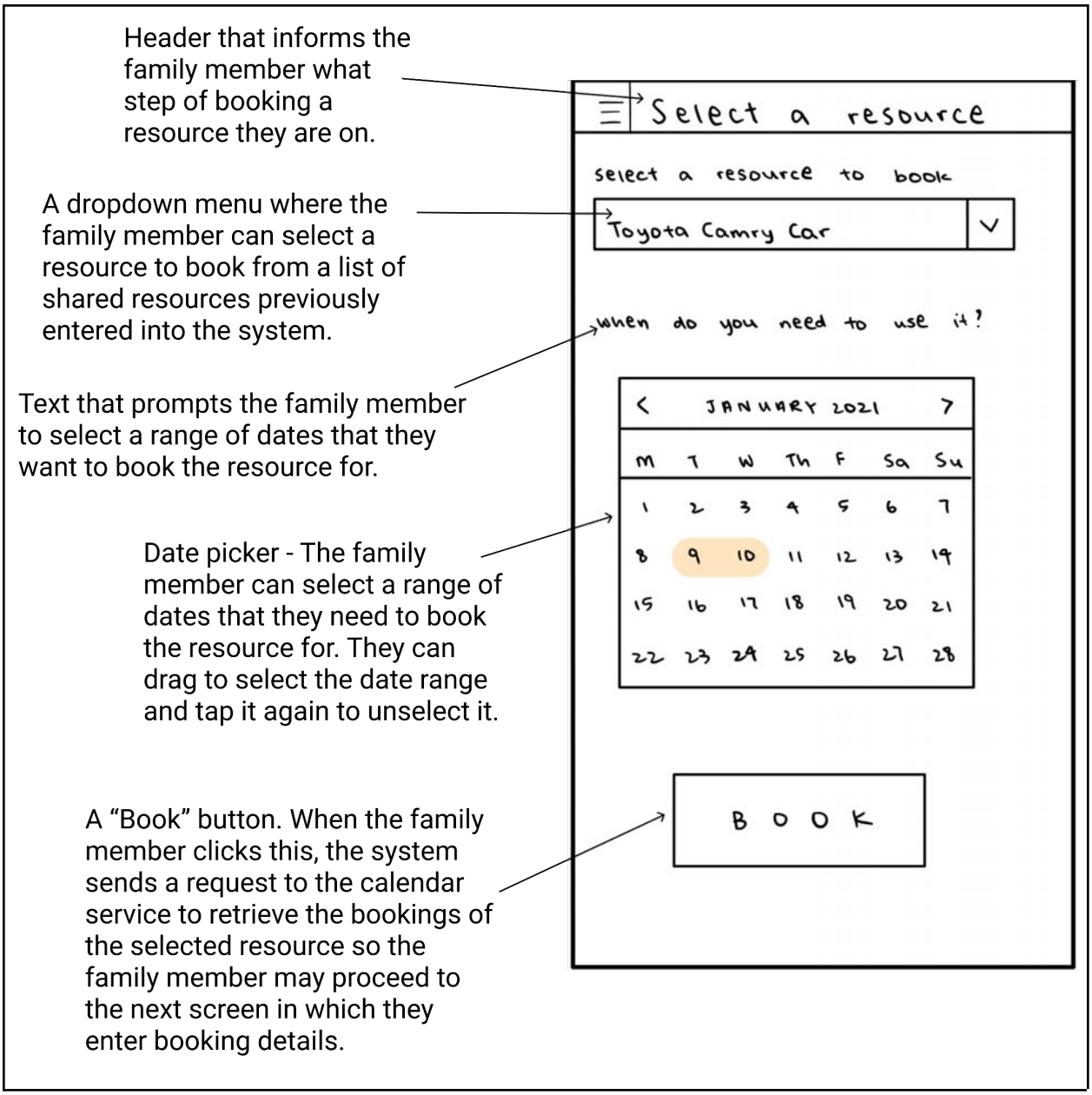

Figure 16: [7] Screen to Select Resource and Usage Date Range

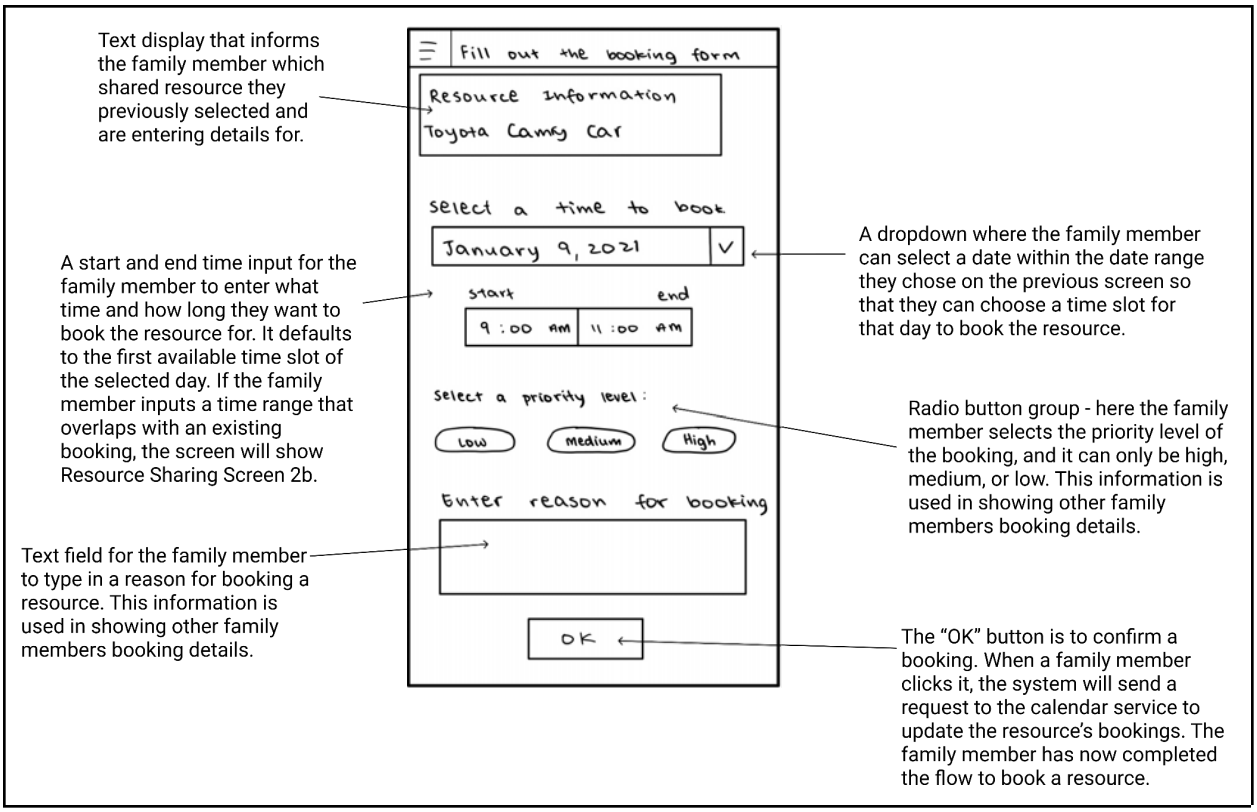

Figure 17: [8] Filling Out Information for the Booking Request of a Resource

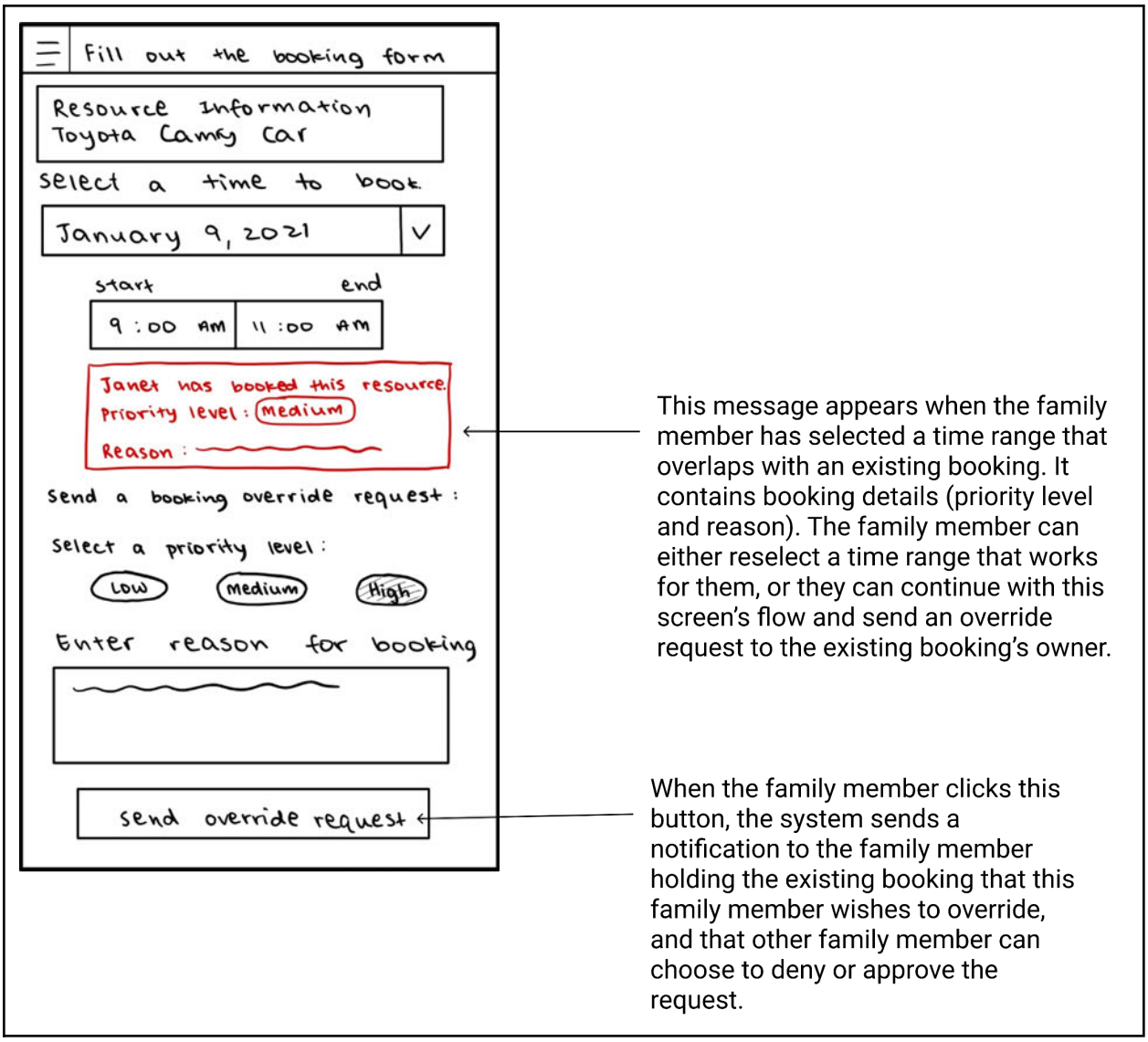

Figure 18: [9] Filling In the Booking Override Request for the Usage of the Resource

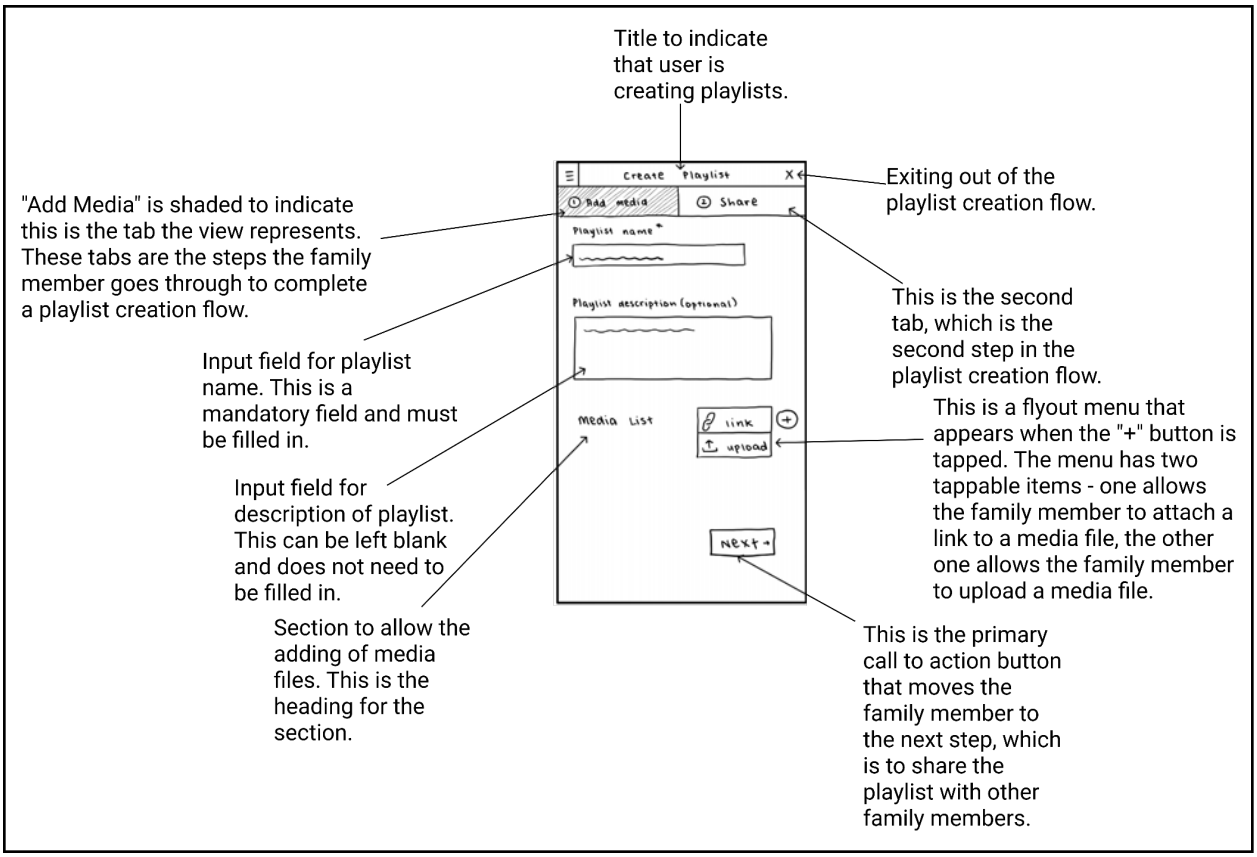

Figure 19: [10] Playlist Creation Screen

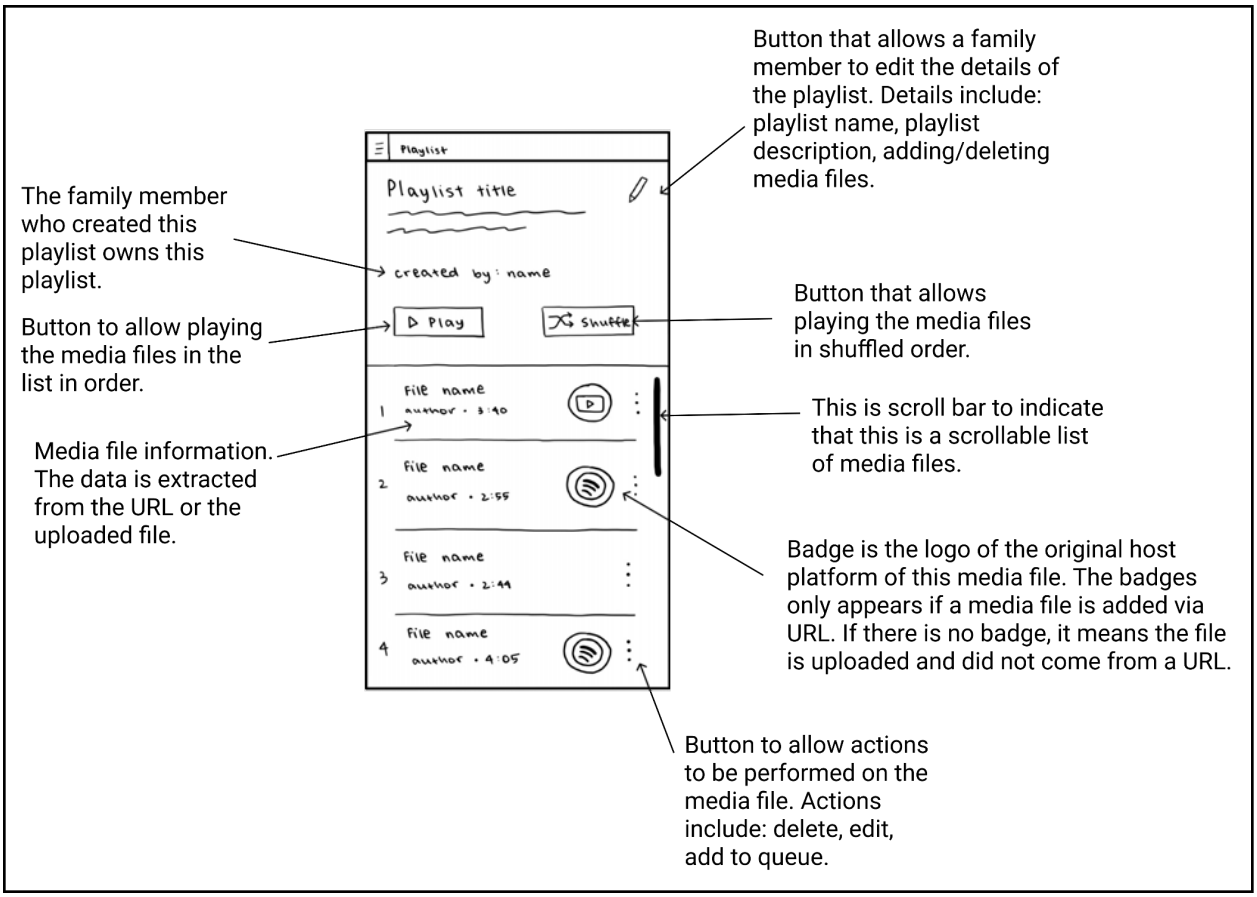

Figure 20: [11] View Playlist Screen

3.2 Scenarios and State Machine Models

This next section contains the scenarios and state machine models for the three use cases.

Scenarios

The scenarios record down the details of the execution paths through a use case by listing the observable actions as back-and-forth interactions between the system and external actors. There are scenarios for each of the three use cases, and each use case includes meta-data about the scenario that builds context for the actions detailed in the elaboration of the diagrams. Specifically, each use case includes a main scenario, four alternative scenarios and two exception scenarios.

Use Case 1: Plan a Family Activity

Scenario Name and Number: Plan a Family Activity (PFA1)
Trigger: A family member clicks “Start Activity” to start a plan for a family activity on the system
Preconditions:

- Family member has an account on the system and is logged into that account
- Other family members have accounts on the system
- The system has access to all family members’ calendars from the integration with the calendar service
- The family members have preset different family activity types and associated tasks with each type

Stakeholders:

- Family members
- Technical experts from the calendar service

Postcondition:

- A family activity has been planned with a name of the activity, activity description, a scheduled date and time, location, activity type and assigned tasks
- The family activity is reflected in each invited family member’s calendar on the calendar service
- Invitees receive a notification, through the system, about the scheduled activity and their assigned tasks

A Family Member (Host)	**System**	**Other Family Members (Invitees)**	**Calendar Service**
1. Clicks “Start Activity” to start a plan for a family activity			
	2. Generates a new family activity page with all family members selected as the default for invitees		
3. Clicks “Poll” as the method of determining the family activity			
	4. Generates a new page that allows the host family member to create a poll		

5. Enters poll options (in the form of activity names and activity descriptions) and a deadline to vote. Then, clicks “Send Poll”			
	6. Sends the poll to all invitees in the form of a notification		
		7. Selects most preferred activities among the options and sends the votes to the poll by clicking “Submit Votes”	
	8. Calculates the poll option with the most votes and presents the result to the host once the deadline to vote has passed		
9. Opens the poll result			
	10. Generates a new page that allows the host to schedule the family activity		
11. Picks a range of days on the date picker component, specifies an intended duration (in hours) for the activity and selects the			

time(s) of day the activity should occur on			
	12. Sends a request to fetch each invitee's availability for the selected days, during the specified time(s) of day and for the intended duration		
			13. Returns the invitees' availabilities based on the selected days, for the specified time(s) of day and for the intended duration
	14. Takes the availabilities that are sent back and presents them as options to the host family member		
15. Selects a time slot out of the presented options. Then, clicks "Next"			
	16. Generates a new page that allows the host to fill in the details of the family activity		
17. Inputs the location of the activity and selects the activity type from a list of pre-set family			

activity types			
	18. Generates a list of unassigned tasks based on the selected family activity type		
19. Assigns invitees to each task in the list and clicks “Send Invites” to finalise the family activity			
	20. Sends a notification to all invitees and the calendar service with details about the activity, including the name of the activity, the activity description, the scheduled date and time, the location, the activity type, and their individual assigned tasks		
		21. Receives a notification about the name of the activity, the activity description, the scheduled date and time, the location, the activity type, and their assigned tasks	
Alternative 1 - The family member does not send a poll for activity options			
A1.3. Clicks “Manually Enter Activity” to input a family activity without sending a poll			

	A1.4. Generates fields for the host to add the name of the family activity and a description of the family activity		
A1.5. Inputs the name of the family activity and the description of the family activity in the fields. Then, clicks “Next”			
	A1.6. Go to 10		
Alternative 2 - The family member does not check for other family members’ availability before scheduling a date and time			
A2.11. Toggles the option to turn the calendar service off to manually select a specified date and time for the activity			
	A2.12. Presents a date picker and a time selector		
A2.13. Selects a date and time for the family activity and clicks “Next”			
	A2.14. Go to 16		
Alternative 3 - The family member adds or removes family members from the activity			
A3.3. Clicks “Change List of Invitees” to select which specific family members to invite to the activity			
	A3.4. Generates an editable list of all family members in the system		
A3.5. Edits the list to specify which family			

members to invite to the activity			
A3.6. Go to 3			
Alternative 4 - The family member does not use a preset family activity type			
A4.17. Inputs the location of the activity and clicks “Add Activity Type” to create a custom activity type			
	A4.18. Generates a blank field for the custom activity type and a blank list of tasks		
A4.19. Inputs the name of the activity type and fills in the list of tasks			
	A4.20. Go to 18		
Exception 1 - Other family members do not respond to the poll for family activity options by the deadline			
		E1.7. Does not respond to the poll by the deadline	
	E1.8. Generates a notification to inform the host that no responses were received for the poll and gives the host the option to manually input an activity to proceed with		
E1.9. Receives the notification, opens it, and manually inputs an activity for the family (name and activity description)			
	E1.10. Go to 10		
Exception 2 - The calendar service fails to return the other family members’			

availability			
			E2.13. Fails to return the invitees' availabilities and sends this status to the system
	E2.14. Receives the failure status and generates a message to inform the host of the failure, and gives the host the option to manually specify a date and time		
E2.15. Receives the message, and manually selects a date and time for the activity. Then, clicks "Next"			
	E2.16. Go to 16		

Use Case 2: Coordinate the Usage of Shared Resources

Scenario Name and Number: User books a shared resource (CUSR1)
Trigger: Family member clicks "RESOURCES" on the menu page to begin booking a resource on the system
Preconditions:

- Family member has an account and is logged into the system
- Family members use the system whenever they want to book a shared resource, so the bookings are always up-to-date.
- The system defines a priority level of High, Medium, or Low for each booking

Stakeholders:

- Family members
- Technical experts from the calendar service

Postcondition:

- Family members have booked a resource and the changes are reflected in both the resource's booking calendar and that family member's calendar

Family Member	System	Other Family Members	Calendar Service
1. Selects the			

desired resource and date range to book			
	2. Sends a request to the calendar service to retrieve the availabilities of the selected resource and date range		
			3. Returns the availabilities of the selected resource
	4. Prompts the family member to select a time frame to book the resource		
5. Selects an available time frame to book the resource			
	6. Prompts the family member to complete the booking request		
7. Completes the booking request details, including the booking reason and priority level, and clicks “Submit” to book the resource during the selected time range			
	8. Sends a request to the calendar service to update the family member and resource calendars with the selected time range, reason for booking, and priority level		
			9. Updates the family member and the

			resource calendar with the booking time, reason, and priority level
Alternative 1 - The resource the family member wants to book is unavailable			
A1.5 Selects an unavailable or overlapping time frame for a booked resource			
	A1.6 Displays existing booking details, including the family member who owns booking, reason, and priority level, and prompts family member to send a booking override request		
A1.7 Fills in booking override request details, including booking reason and priority level, and clicks “Send”			
	A1.8 Notifies existing booking owner of the shared resource with its details through in-app notification service and prompts them to approve or deny it		
		A1.9 Approves the override request	
	A1.10 Cancels the existing booking and sends request to calendar service to update calendars with this change		
			A1.11 Updates resource’s and family member’s

			calendar with the cancelled booking
	A1.12 Go to 8		
Alternative 2 - The family member wants to add a new shared resource to the system			
A2.1 Chooses option to add a resource			
	A2.2 Prompts user to input details regarding the new shared resource, such as resource name		
A2.3 Inputs the resource name			
	A2.4 Saves the new shared resource and brings user back to the select resource step		
A2.5 Go to 1			
Alternative 3 - Another family member denies a booking override request			
		A3.A1.9 Denies override request	
	A3.A1.10 Notifies the family member, who requested the override of the request, of the rejection through an in-app notification service		
A3.A1.11 Go to 5			
Alternative 4 - Family member wants to change the date range to view availabilities for a selected resource			
A4.5 Selects a new date range to view resource booking availabilities			
	A4.6 Go to 2		

Exception 1 - Another family member does not respond to override request			
		E1.A1.9 Ignores override request	
	E1.A1.10 Waits 24 hours since the override request is sent, then notifies the family member who sent override request that it was not responded to if no response is received within that time		
E1.A1.11 Go to 5			
Exception 2 - Calendar service is down and fails to respond			
			E2.3. Returns an error message where the server status is not available
	E2.4. Prompts family member to enter booking details including time, reason, and priority levels		
E2.5. Completes the booking request details, including the booking reason and priority level, and submits the request to book the resource during the selected time range			
	E2.6. Notifies all family members of booking request through in-app notification service		
		E2.7. All family members acknowledge the booking request	

	E2.8. Notifies family member who booked the resource that a booking has been made through in-app notification service		
	E2.9 Go to 8		

Use Case 3: Collaborate on Shared Media Playlists

Scenario Name and Number: Create a Shared Media Playlist (CSMP 1)
Trigger: A family member clicks on the “Create Playlist” button to create a new playlist on the system
Preconditions:

- Family member that creates a playlist has an account registered on the system and is logged in
- The family member knows the emails of family members (those who are not registered on the system) they want to share the playlist with
- Online Audio/Video Media platform offers shareable links to individual media files on their platforms
- The system has permission to access files on the family member’s local device storage

Stakeholders:

- Family members
- Technical experts of audio/video platforms

Postcondition:

- Audio and/or video media playlist is successfully created and can be viewed and edited by its creator and editors
- Family members, who are invited to collaborate on the playlist, receive a notification, through either system or email, about the playlist that is shared with them

A Family Member	**System**	**Online Audio/Video Media Services**	**Email Service**	**Other Family Members**
1. Initiates the creation flow of a playlist by clicking the “Create Playlist” button				

	2. Generates input fields to input the name and description of the playlist			
3. Inputs the name and description of the playlist				
	4. Displays options (upload or paste URL) to allow the family member to add audio or video media files to the playlist			
5. Pastes shareable links of the desired online media files				
	6. Sends a request to fetch online media files from the pasted links			
		7. Approves the request and sends back the confirmation status with the requested media files		
	8. Displays information about media files in the playlist			
9. Clicks “Next” button to proceed to the next step				

in playlist creation flow				
	10. Provides a dropdown of potential family members to share the playlist with			
11. Selects all family members from the list to share the playlist to and click “Complete and Create” button to complete creating the playlist				
	12. Creates notifications for selected family members and sends invite notifications to them using the in-app notification service			
				13. Selected family members receive the notification. They click the notification to access the playlist
	14. Displays list view of items in the playlist. Also displays an ellipsis button to allow family members to edit the playlist.			

Alternative 1 - Uploading media file from the family member's own device				
A1.5. Chooses to upload media file from own device				
	A1.6. Displays the list of compatible media files in a popup window			
A1.7. Selects the desired file(s) from popup window and confirm upload				
	A1.8. Uploads the media file			
	A1.9. Go to 8			
Alternative 2 - Importing an existing playlist				
A2.5. Pastes a shareable link to existing playlists on other online media platforms				
	A2.6. Go to 6			
Alternative 3 - Family member sends an email invite to family members who are not registered on the system				
A3.10. Does not see a certain family member in dropdown. Click the				

“Invite With Email” button to manually add a family member				
	A3.11. Requests to type in email of the family member in an input field			
A3.12. Inputs the email of the family member in the input field				
	A3.13. Sends invite to the family member’s email			
			A3.14. Displays email that shares the playlist with the other family member	
				A3.15. Opens email and clicks the link in the email to access the playlist
	A3.16 Go to 14			
Alternative 4 - Share to other family members later				
A4.11. Chooses to not share with family members right now, but share it later by clicking “Share				

Later” button				
	A4.12. Saves the playlist as a draft that is not shared with any family members			
	A4.13 Go to 14			
Exception 1 - Unable to fetch online media link from the shareable link				
		E1.7. The online file from the link no longer exists		
	E1.8. Displays “file not found” error and asks the family member to try to add another media file			
E1.9. Go to 5				
Exception 2 - Unable to upload media file from own device				
	E2.A1.6. Displays error about inability to process the file and asks the family member to try again or add another media file			
E2.A1.7. Go to A1.5				

State Machine Models

The state machine models are provided as navigation diagrams for each use case, but they are all connected together in the overarching diagram that models the whole system's behaviour. The states are modelled as UI screens, some corresponding to the sketches in Section 3.1, and others are just references to screens that have not been sketched out in this document. The transitions represent how a family member would navigate between the different states. Since some states refer to the UI sketches, the same screen numbers correspond to the sketches in the table in Section 3.1.

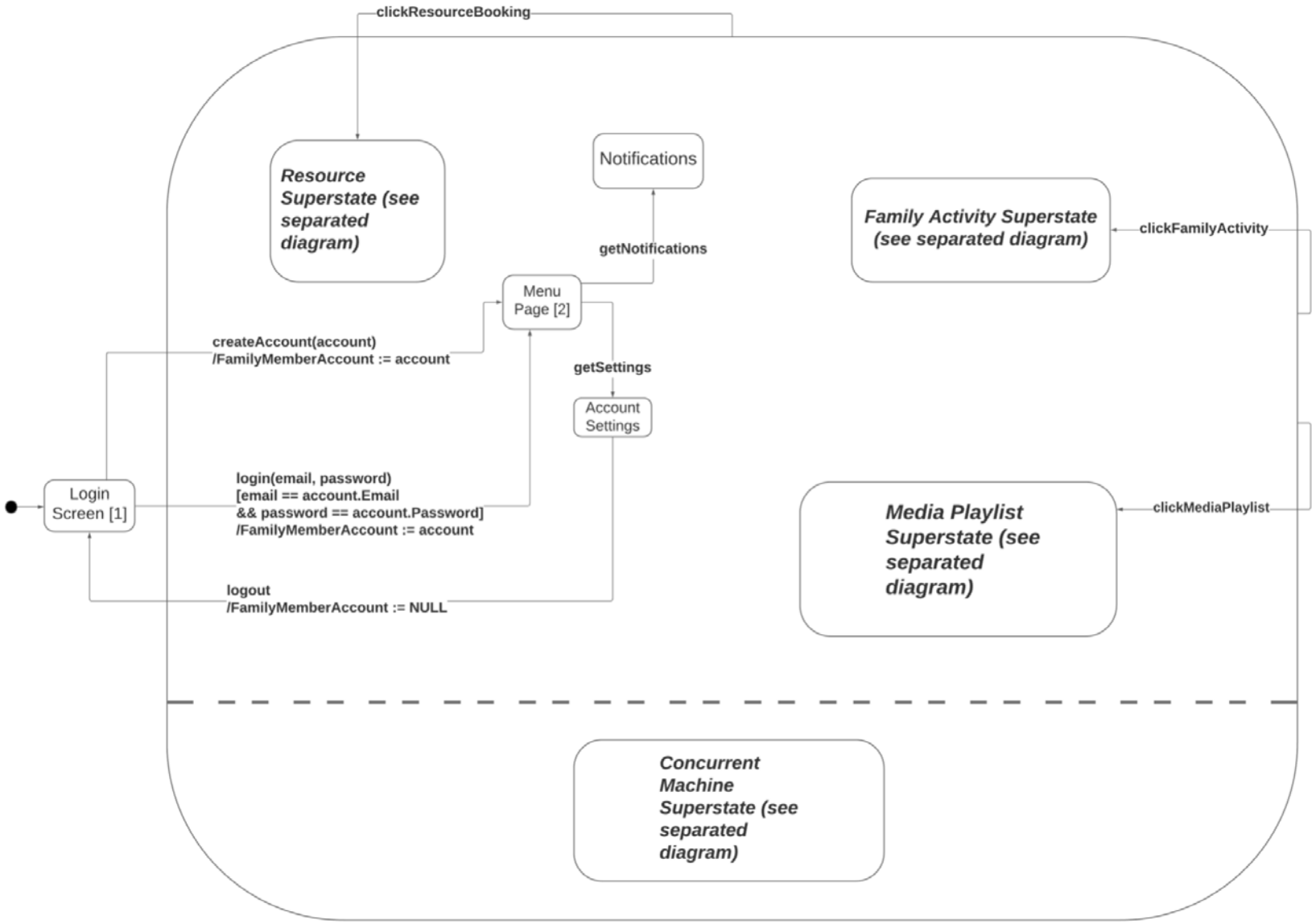

Figure 21: Overarching Navigation Diagram

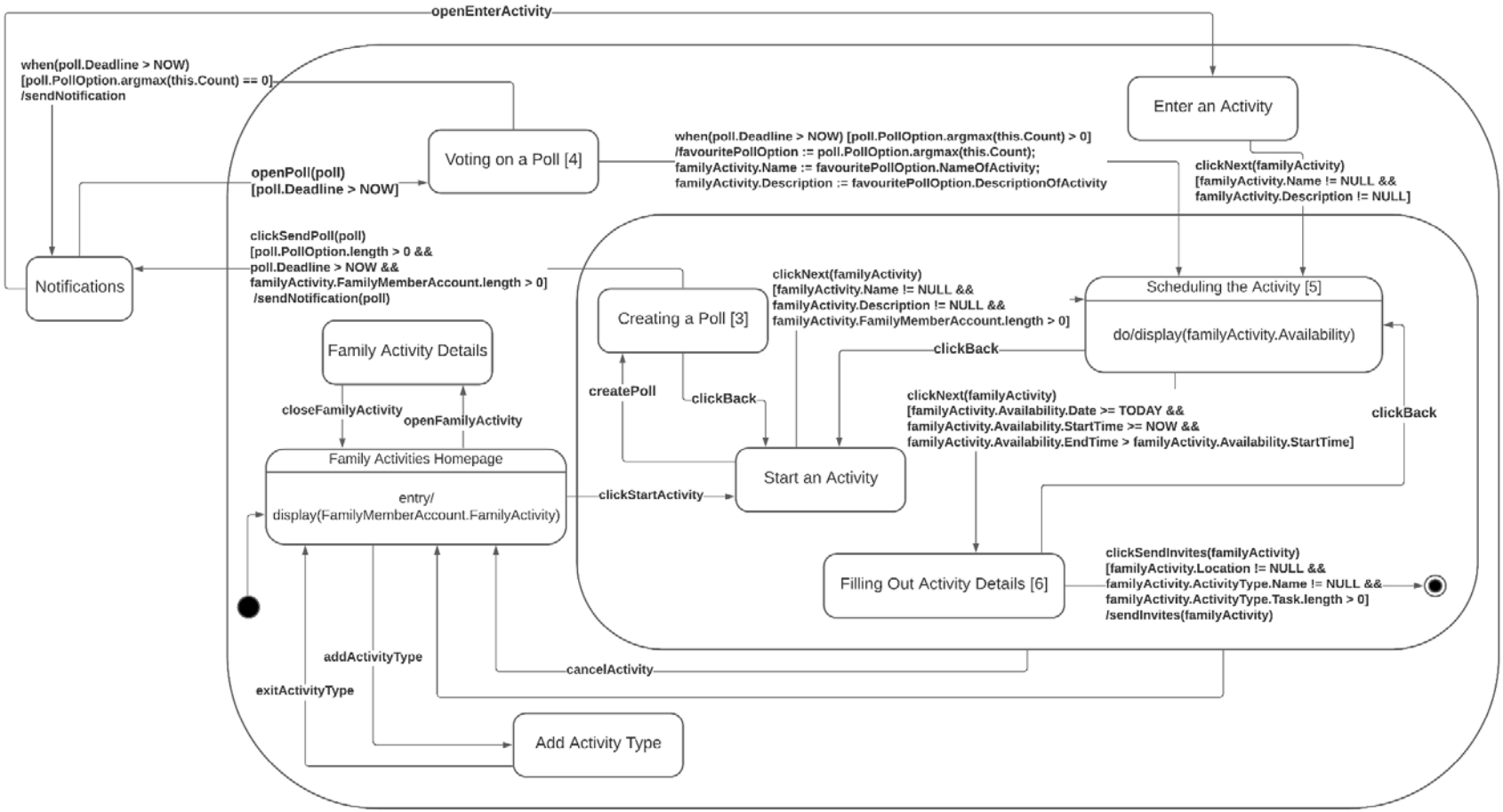

Figure 22: Navigation Diagram for Planning a Family Activity Use Case

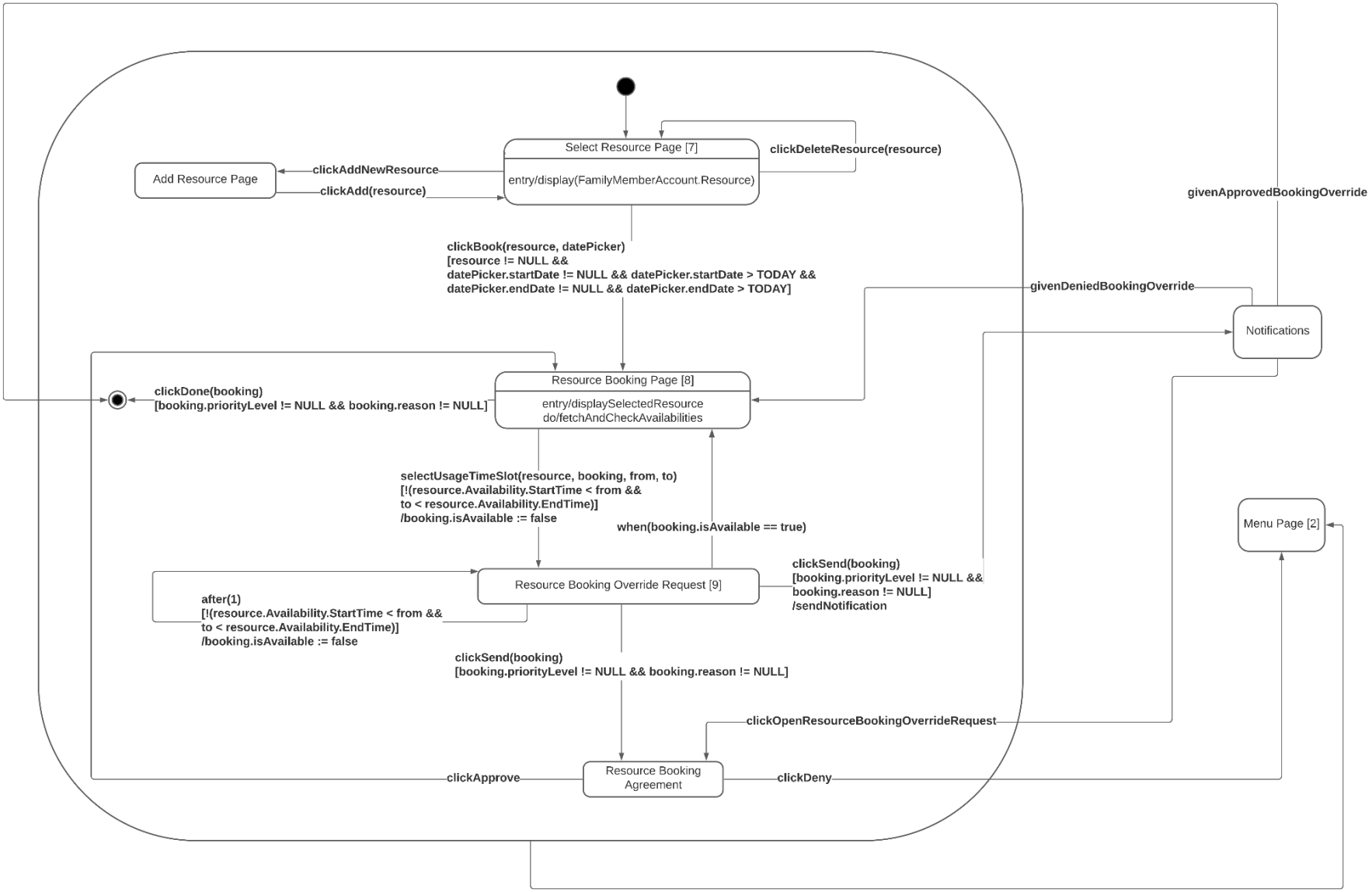

Figure 23: Navigation Diagram for Coordinate the Usage of Shared Resources Use Case

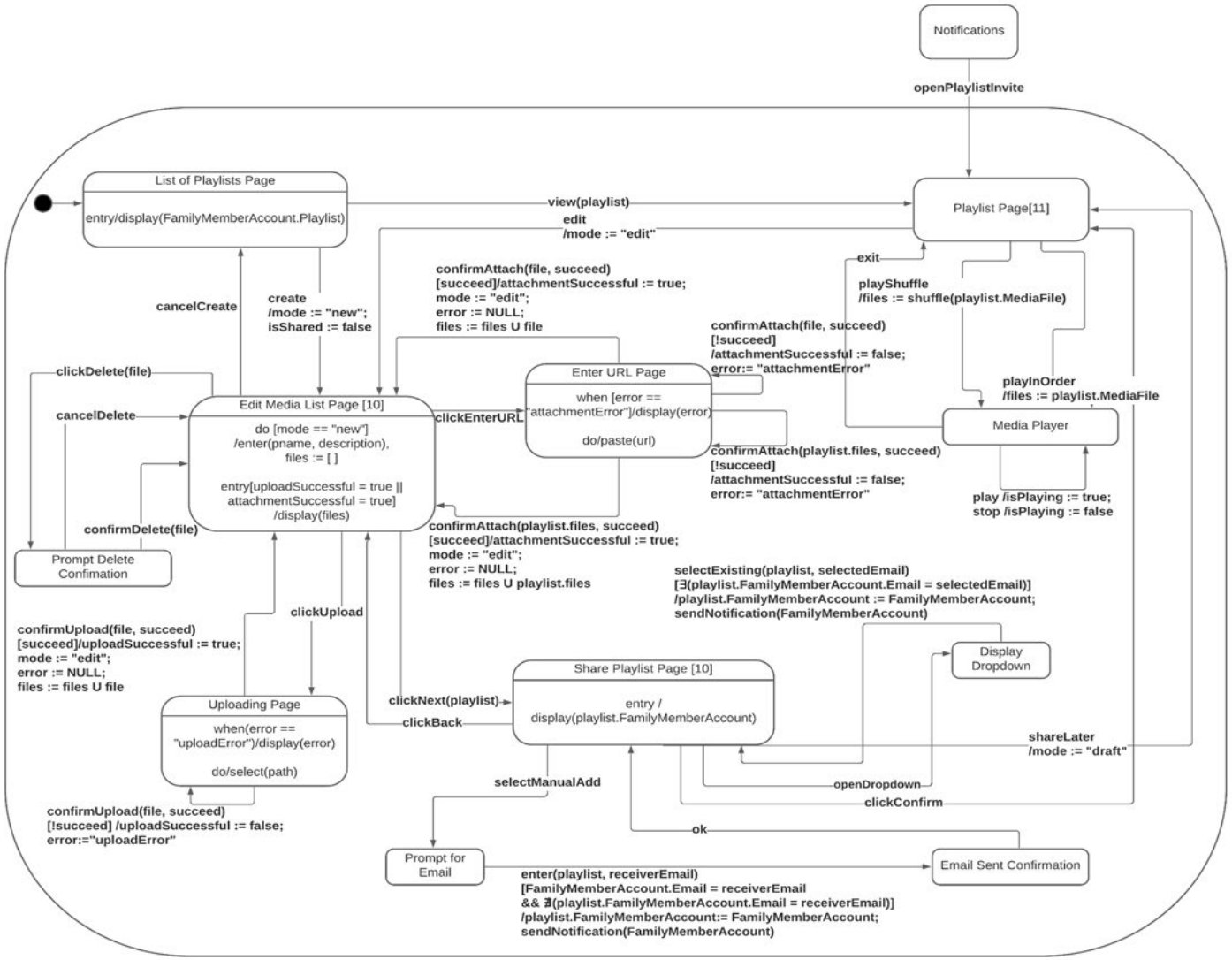

Figure 24: Navigation Diagram for Collaborating on Shared Media Playlists Use Case

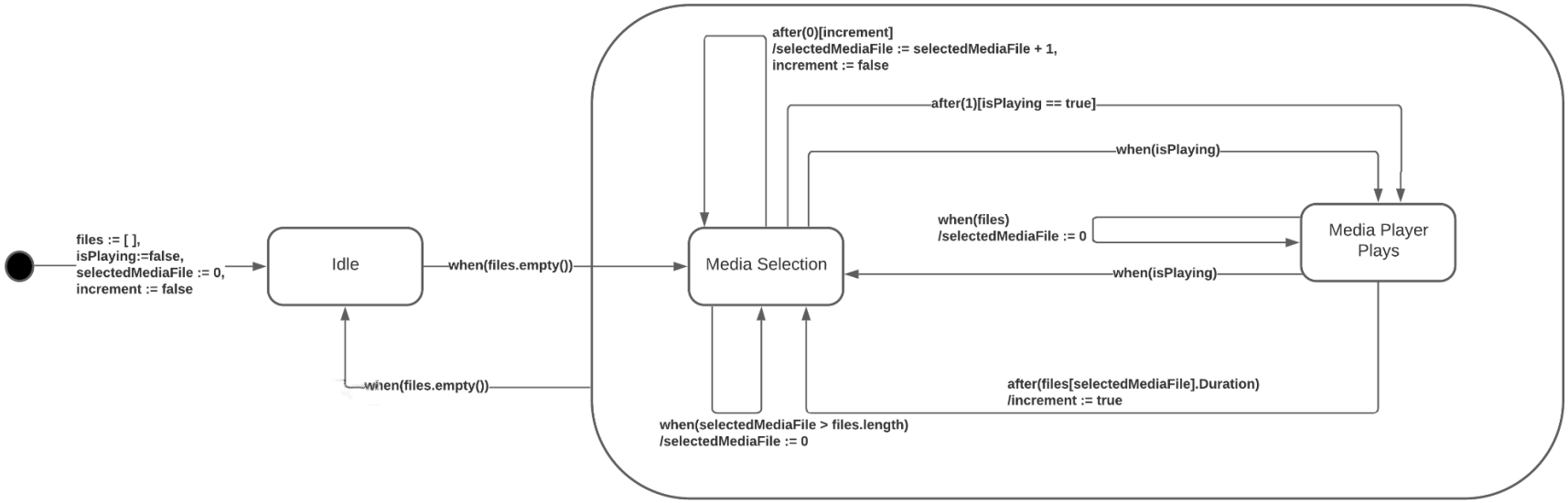

Figure 25: Navigation Diagram for the Media Player Concurrent Machine

Variables Legend

- NOW = current time as tracked by the system
- TODAY = current day as tracked by the system

General (State Legend)

Login Screen [1]
An interface of the application that provides the input fields for an email and a password for the family members who already have family member accounts on the system. For the family members who do not have an account, it also provides a text link which directs them to create a family member account.

Menu Page [2]
The first screen that the family member accounts land on when they sign in to the application. It displays a centralized management center where the family members can access the use cases such as planning family activities, collaborating on shared playlists, and managing family resources that the application provides.

Account Settings
An interface that displays the log out option for the family member accounts.

Notifications
A state that represents the situation where in-app notifications are sent to family member accounts. The notifications alert about resource booking conflicts, family activity polls, and invitations for collaborating on shared media playlists, among others.

Use Case 1: Plan a Family Activity (State Legend)

Family Activities Homepage
This screen lists all the existing family activities that the family member account is invited to. This is characterized by the display(FamilyMemberAccount.FamilyActivity) action. It also gives family members the option of starting a new family activity.

Family Activity Details
This screen lists the details of a particular family activity that the family member account is invited to. It can be accessed from the Family Activities Homepage.

Add Activity Type
This screen allows the family member to add a family activity type and associated tasks related to that type. This allows the family member to add the preset family activity types that will be used later on in the planning of the family activity.

Start an Activity
This screen is reached when the family member decides to host a family activity. On this screen, the family member would specify the other family member accounts who are invited to the activity and select a method of deciding what the family activity will be - the latter can be either through a poll or manually entered by the host.

Creating a Poll [3]
After the host decides to use a poll to determine the family activity, a poll is created. The host would specify the deadline for voting and a list of poll options (each one includes the name of a potential family activity and a description of the activity), and then sends the poll to the invited family member accounts.

Scheduling the Activity [5]
After the family activity is determined, the family member who hosts the activity selects a date range, an intended duration and at least one time of day for the activity. The system constantly fetches the up-to-date common availabilities of the participating family members, based on the specified criteria, and through the integration with the calendar service (this is done through the display(familyActivity.Availability) activity).

Filling Out Activity Details [6]
After the date and time of the family activity is selected, the host fills out the remaining details (i.e. location, activity type, tasks to be assigned) of the upcoming family activity.

Voting on a Poll [4]
After the family member who hosts the family activity decides to open a poll for the potential family activities, the family members who are invited to participate get a notification for the poll in which they can choose their desired family activities among all the options provided on the poll. The poll is closed when the deadline to vote has passed.

Enter an Activity
In the event that no family members submit their votes for the family activity through the poll, the host of the family activity will receive a notification and the option to manually enter the family activity name and description. This screen allows the host to do that.

Use Case 2: Coordinate the Usage of Shared Resources (State Legend)

Add Resource Page
A form interface that allows a family member to enter information about a shared resource to the family's shared resource inventory.

Select Resource Page [7]
An interface that shows an inventory of shared resources of the family. Here, the family member can select a resource and date range for booking.

Resource Booking Page [8]
An interface that requires the necessary information about the booking of a shared resource. The family member needs to choose a priority level, a time slot for the usage, as well as specifying the reason for using the resource. Once the family member chooses the usage time slot, the system checks if the availabilities of the resource contains the chosen usage time slot, and assigns corresponding value to isAvailable.

Resource Booking Override Request [9]
An interface that is a variation of Resource Booking Page [8] that only shows when a conflict for using any shared resource occurs. However, the usage time slot on resource override request is dynamic, and every time the family member modifies the usage time slot, the system checks if the availabilities of the resource contains the chosen usage time slot, and assigns the corresponding value to isAvailable.

Resource Booking Agreement
Family members who book a shared resource prior to other members who want to book the same resource on overlapping time slots receive the notifications for conflict. The notifications direct to an agreement screen that allows the family members to view the priority level and reason from other members, and to decide to approve or deny overriding their earlier booking of the resource.

Use Case 3: Collaborate on Shared Media Playlists (State Legend)

List of Playlists Page
An interface that displays a list of playlists that the family member account has access to.

Enter URL Page
A page where the family member needs to paste a valid URL to the media file.

Edit Media List Page [10]
The first part of the playlist creation screen which requires the family member to enter the name and the description of the newly created playlist. The family member can continuously add media files to the playlist until they are done creating the playlist.

Share Playlist Page [10]
The second part of the playlist creation screen which allows the family member to share the playlist with other family members. This step checks if the playlist is shared, and this status will be reflected on the page.

Playlist Page [11]
An interface of a single playlist that displays the name, description, list of media files, and if the list is shared.

Uploading Page
A transition page that shows when family members upload a local media file to the application. It checks if any error occurs during uploading, and displays error messages whenever there are any problems with uploading.

Prompt Delete Confirmation
This is a confirmation prompt that appears when a family member wants to delete a media file.

Prompt for Email
Preview of the invitation email to invite family members to collaborate on the media playlists, and the confirmation button to send the email.

Email Sent Confirmation
A confirmation screen that communicates that invitations for collaborating on the playlist are sent.

Display Dropdown
A dropdown that consists of a list of family members who have an account registered with the system with whom the playlist can be shared.

Media Player
A service that plays media files, where the state can be paused or continued for playing.

Concurrent Machine (State Legend)

Idle
A state in which no interaction with the media player has been made yet.

Media Selection
A state that records the selected media file that is being played, and determines if the application should play the next file based on if there is an increment in the files or if the family member has reached the end of the playlist.

Media Player Plays
A state that shows if the media player is playing a media file.

General (Transition Legend)

login(email, password)
[email == account.Email && password == account.Password]
/FamilyMemberAccount := account
The family member logs into their family member account with an email and a password on the Login Screen [1]. This action checks if the input email and password pair matches what is stored in the family member account object and brings the family member to the Menu Page [2].

logout
/FamilyMemberAccount := NULL
The family member logs out of their family member account from the Account Settings page. This action sets the FamilyMemberAccount to NULL and brings the family member back to the Login Screen [1].

createAccount(account)
/FamilyMemberAccount := account
The family member creates a new account from the Login Screen [1]. This action sets the FamilyMemberAccount to the newly created account and brings the family member to the main Menu Page [2].

getSettings
From the Menu Page [2], the family member can access their account settings from the downwards arrow symbol beside the “Family Centre” headline. They are able to logout of their family member account from here as well.

getNotifications
From the bell icon on the Menu Page [2], the family member is able to view a list of all their notifications on the Notifications screen.

clickFamilyActivity
Family members are able to click the “Family Activity" option to access the family activity superstate through the hamburger menu from anywhere in the application or directly from the Menu Page [2]. The use case’s start state is the Family Activities Homepage screen.

clickResourceBooking
Family members are able to access this use case superstate through the hamburger menu from anywhere in the application or directly from the Menu Page [2]. The use case’s start state is the Select Resource Page [7] screen which provides a list view of the resource inventory a family owns as well as a Date Picker for the family member to select a date range.

clickMediaPlaylist
Family members are able to access this use case’s superstate through the hamburger menu from anywhere in the application or directly from the Menu Page [2]. The use case’s start state is the List of Playlists Page which provides a list view of all the playlists a family member owns or has been invited to collaborate on.

openFamilyActivity
Family members are able to view the details of a particular family activity by clicking on “Open” listed next to that family activity on the Family Activities Homepage.

closeFamilyActivity
To close the details of this family activity, family members are able to click “Close” on the page, and then they will navigate back to the Family Activities Homepage.

addActivityType
If a family member wants to add a preset family activity type, they can do so by clicking “Add Activity Type” on the Family Activities Homepage. They will be navigated to the Add Activity Type screen to add the name and associate tasks of an activity type.

exitActivityType
Once the family member has finished adding their activity type, then they can exit the page by clicking on “Exit” and returning back to the Family Activities Homepage.

clickStartActivity
If the family member would like to start a new activity, this transition will lead them to the Start an Activity screen where they can specify which family members to invite and select a method to determine what activity the family can participate in.

cancelActivity
At any point during the planning of the family activity, the family member can choose to exit the activity creation superstate, cancel the family activity and return back to the Family Activities Homepage. This cancelActivity transition allows them to do this.

clickNext(familyActivity)
[familyActivity.Name != NULL && familyActivity.Description != NULL && familyActivity.FamilyMemberAccount.length > 0]
From the Start an Activity screen, the family member can click “Next” to move on to the next step in this workflow, which is activity scheduling on the Scheduling the Activity [5] screen. During this transition, there is an additional check to ensure the familyActivity.Name, familyActivity.Description and list of invitees (familyActivity.FamilyMemberAccount) are not empty. Additionally, the instance of the activity is passed to the next screen to allow the family member to figure out the scheduling for this particular instance of the activity.

clickNext(familyActivity)
[familyActivity.Availability.Date >= TODAY && familyActivity.Availability.StartTime >= NOW && familyActivity.Availability.EndTime > familyActivity.Availability.StartTime]
From the Scheduling the Activity [5] screen, the family member can click “Next” to move on to the Filling Out Activity Details [6] screen to continue filling out the details of the activity. During

this transition, there is a check to make sure that the familyActivity.Availability.Date, familyActivity.Availability.StartTime and familyActivity.Availability.EndTime are all valid values. Additionally, the instance of the activity is passed to the next screen to allow the family member to fill in the rest of the details for this particular instance of the activity.

clickBack
From the Scheduling the Activity [5] screen, the family member may wish to return to the Start an Activity screen to modify previously entered details. This transition enables family members to return back to that screen.

From the Filling Out Activity Details [6] screen, the family member may wish to return to the Scheduling the Activity [5] screen to modify previously entered details. This transition enables family members to return back to that screen.

From the Creating a Poll [3] screen, the family member may wish to return to the Start an Activity screen to modify previously entered details. This transition enables family members to return back to that screen.

clickSendInvites(familyActivity)
[familyActivity.Location != NULL && familyActivity.ActivityType.Name != NULL && familyActivity.ActivityType.Task.length > 0]
/sendInvites(familyActivity)
Once all activity details have been completed on the Filling Out Activity Details [6] screen, invitations can be sent to all invitees by clicking “Send Invites” on that screen. There is an additional check to make sure that the familyActivity.Location, familyActivity.ActivityType.Name and familyActivity.ActivityType.Task fields are valid. In the sendInvites action, the instance of the activity is passed so the data within the activity can be sent to each of the family members in the familyActivity.FamilyMemberAccount list, as specified in the activity. This transition leads to the end state as the use case has now been completed.

createPoll
From the Start an Activity page, the family member may choose to initiate a poll to determine what type of family activity their family members may be interested in. The createPoll transition brings the family member to the Creating a Poll [3] screen where the poll details can be set up.

clickSendPoll(poll)
[poll.PollOption.length > 0 && poll.Deadline > NOW &&
familyActivity.FamilyMemberAccount.length > 0]
/sendNotification(poll)
Once the poll has been set up, a notification must be sent to the list of invitees asking them to submit their votes to the family activity poll. The poll is passed as a parameter in this transition. Furthermore, the poll.PollOption.length, poll.Deadline and familyActivity.FamilyMemberAccount fields are checked for validity before sending the notification through the sendNotification action.

openPoll(poll)
[poll.Deadline > NOW]
If invited to partake in a family activity poll, the family member will be notified via the in-app notification service. All notifications can be viewed from the Notifications screen. Clicking on the poll notification will open the poll for family members to make their voting selections on the Voting on a Poll [4] screen. Family members can only open this poll if the deadline has not yet passed.

when(poll.Deadline > NOW)
[poll.PollOption.argmax(this.Count) > 0]
/favouritePollOption := poll.PollOption.argmax(this.Count);
familyActivity.Name := favouritePollOption.NameOfActivity;
familyActivity.Description := favouritePollOption.DescriptionOfActivity
Once the deadline has passed, and at least one vote has been submitted for a pollOption, then the system will calculate the pollOption with the highest Count value. That option will be the family activity that the family members will partake in (favouritePollOption). Additionally, during this transition, the familyActivity.Name is set to the favouritePollOption.NameOfActivity and the familyActivity.Description is set to the favouritePollOption.DescriptionOfActivity. After this, the family member moves on to the Scheduling the Activity [5] screen.

when(poll.Deadline > NOW)
[poll.PollOption.argmax(this.Count) == 0]
/sendNotification
Once the deadline has passed, and there are no votes submitted for a pollOption, then the system will send a notification to the host of the family activity. This notification will allow the host family member to manually input a name and description for the family activity.

openEnterActivity
The host family member can open the notification that tells them that no votes were submitted for the poll. They will be redirected to the Enter an Activity screen where they can manually enter values for familyActivity.Name and familyActivity.Description.

clickNext(familyActivity)
[familyActivity.Name != NULL && familyActivity.Description != NULL]
Once the host family member has entered the familyActivity.Name and familyActivity.Description, they can click “Next” to proceed to the scheduling step on the Scheduling the Activity [5] screen.

Empty transition from the superstate for planning the family activity back to the Family Activities Homepage
Once the family member has finished planning the family activity (they have reached the termination state of the superstate for planning the activity), then they will be automatically redirected back to the Family Activities Homepage.

clickAddNewResource
From the Select a Resource Page [7], family members may wish to add a new family resource (i.e. a new vehicle, an additional TV, etc.). This will bring the family member to the Add Resource Page where they are able to input the new resource's details such as the resource name, description, and more.

clickAdd(resource)
Once the resource details have been filled out on the Add Resource Page, the "Add" button can be selected. This transition will lead the family member back to the Select a Resource Page [7] with the newly added resource included.

clickDeleteResource(resource)
From the Select a Resource Page [7], the family member may wish to delete an existing resource. This transition will lead the family member back to the Select a Resource Page [7], this time with the deleted resource removed.

clickBook(resource, datePicker)
[resource != NULL && datePicker.startDate != NULL && datePicker.startDate > TODAY && datePicker.endDate != NULL && datePicker.endDate > TODAY]
Once a resource has been selected from the Select a Resource Page [7], the family member will need to input a date range for which they want to borrow the resource for. When they click on the "Book" button, this transition will bring the family member to the Resource Booking Page [8] where booking information details can be filled out. The selected startDate and endDate must not be NULL and must be a future date.

selectUsageTimeSlot(resource, from, to)
[!(resource.Availability.StartTime < from && to < resource.Availability.EndTime)]
/booking.isAvailable := false
When a family member makes a change to the start time, from, and end time, to, that they want to book a resource for, on Resource Booking Page [8], and the selected time range is not contained in the Resource's Availability time range, the Booking's isAvailable attribute is set to false and the system brings the family member to the Resource Booking Override Request [9].

clickDone(booking)
[booking.priorityLevel != NULL && booking.reason != NULL]
On the Resource Booking Page [8], details such as the date and time, priority level and reason for booking are filled by the family member prior to finalising the booking. Once the "Done" button is clicked, the system updates the calendars by making a request to the calendar service, which means a family member has successfully booked the resource for their selected date and time range This leads to the end state in this superstate.

when(booking.isAvailable == true)
On the Resource Booking Override Request [9], a family member can change the start time and end time that they want to book for, which updates the Booking's isAvailable attribute. Once the selected time range is available, the family member is shown the Resource Booking Page [8].

after(1)
[!(resource.Availability.StartTime < from && to < resource.Availability.EndTime)]
/booking.isAvailable := false
While the family member is on the Resource Booking Override Request [9], the system will constantly check if the selected time range is available, and will set Booking.isAvailable to false so that the screen will remain the same.

clickSend(booking)
[booking.priorityLevel != NULL && booking.reason != NULL]
Should a family member wish to override another family member's resource booking, they will be led to the Resource Booking Override Request [9]. Once the request is sent, the priority level and reason of the requestor is sent to the Resource Booking Agreement, which the original booker of the resource can view and subsequently approve or deny the request.

clickSend(booking)
[booking.priorityLevel != NULL && booking.reason != NULL]
/sendNotification
When a family member clicks 'Send' for an override request, it sends a notification to the existing booking owner through Notifications.

clickOpenResourceBookingOverrideRequest
When the family member opens their notifications and sees that they have a request from another family member to override their existing booking, they can click a button to view the override request's details in the Resource Booking Agreement.

clickApprove
When a family member who has been sent an override request clicks on "Approve" for the override request, this cancels their existing booking, and it brings them to the Resource Booking Page [8] so they can schedule another booking for the resource.

clickDeny
When a family member who has been sent an override request clicks on "Deny" for the override request, nothing happens to their existing booking and they are simply brought to the Menu Page [2].

givenDeniedBookingOverride
When a family member who sent an override request has been denied, they will see this in their notifications, and will be brought to the Resource Booking Page [8] to schedule another booking.

givenApprovedBookingOverride
When a family member who sent an override request has been approved, they will see this in their notifications, and this brings them to successfully book a resource as they require.

Empty transition from the superstate for booking a resource back to Menu Page [2]
Once a family member is finished booking a resource, then they will be automatically redirected back to the Menu Page [2].

Use Case 3: Collaborate on Shared Media Playlists (Transition Legend)

view(playlist)
From the List of Playlists Page, the family member may view an individual playlist and be brought to the Playlist Page [11] where they can view the selected playlist's name, description and files.

edit
/mode := "edit"
From the Playlist Page [11], the family member may edit the playlist. This transition brings the family member to the Edit Media List [10] page and sets the mode to the string edit.

openDropdown
From the Share Playlist Page [10], the family member can view a dropdown of family members that the playlist can be shared with.

playShuffle
/files := shuffle(playlist.MediaFile)
On the Playlist Page [11], a family member may wish to shuffle the order of the media files in their playlist when they play it. This transition brings the family member to the Media Player screen and sets all media files to shuffle.

playInOrder
/files := playlist.MediaFile
Instead of shuffle play, a family member may wish to play their media files in order. From the Playlist Page [11], a family member can play their media in order. This transition brings family members to the Media Player and sets files to play in order.

exit
While using the Media Player, the family member may wish to leave the Media Player and return to the Playlist Page again. This is completed by the exit transition which brings the family member back to the Playlist Page [11].

create
/mode := “new”;
isShared := false
From the List of Playlists Page, the family member may wish to create a new playlist. This transition will bring the family member to the Edit Media List Page [10] where the details of their new playlist can be inputted. This transition also sets the mode to the string new.

cancelCreate
If the family member no longer wants to create a new playlist, they may click cancel at any time and be brought back to the List of Playlists Page.

clickEnterURL
From the Edit Media List Page [10], the family member may upload a media file via a URL link. Selecting this option brings the family member to the Enter URL Page.

confirmAttach(file, succeed)
[!succeed]
/attachmentSuccessful := false;
error := “attachmentError”
The URL attachment of a media file is unsuccessful, the family member is brought back to the Enter URL Page and is presented with an error message.

confirmAttach(file, succeed)
[succeed]
/attachmentSuccessful := true;
mode := “edit”;
error := NULL;
files := files U file
Once the appropriate URL for the media file has been successfully attached, the attachmentSuccessful boolean is set to true and the file is added to the playlist.

confirmAttach(playlist.files, succeed)
[!succeed]
/attachmentSuccessful := false;
error:= "attachmentError"
The URL attachment of all the media files in an existing playlist from the adjacent media platform is unsuccessful, the family member is brought back to the Enter URL Page and is presented with an error message.

confirmAttach(playlist.files, succeed)
[succeed]
/attachmentSuccessful := true;
mode := "edit";
error := NULL;
files := files U playlist.files
Once the appropriate URL for all the media files in an existing playlist from the adjacent media platform have been successfully attached, the attachmentSuccessful boolean is set to true and the file is added to the playlist.

clickUpload
From the Edit Media List Page [10], the family member may choose to upload a media file from their own device. Selecting this option brings the family member to the Uploading Page.

confirmUpload(file, succeed)
[!succeed]
/uploadSuccessful := false;
error := “uploadError”
The file upload is unsuccessful, the family member is brought back to the Uploading Page and is presented with an error message.

confirmUpload(file, succeed)
[succeed]
/uploadSuccessful := true;
mode := “edit”;
error := NULL;
files := files U file
Once the appropriate media file has been successfully uploaded, the uploadSuccessful boolean is set to true and the file is added to the playlist.

clickDelete(file)
From the Edit Media List Page [10], the family member can delete a media file by clicking on the delete control. This leads the family member to the Prompt Delete Confirmation pop-up where the family member can confirm or cancel the media file deletion.

cancelDelete
If the family member no longer wants to delete the media file, they may click “Cancel” at any time and be brought back to the Edit Media List Page [10].

confirmDelete(file)
Confirming the deletion will delete the media file and bring the family member back to the Edit Media List Page [10].

clickNext(playlist)
From the Edit Media List Page [10], if the family member selects the “Next” button, they will be brought to the Share Playlist Page [10] where they can share the playlist with other family members.

clickBack
From the Share Playlist Page [10], if the family member selects the “Back” control, they will be brought back to the Edit Media List Page [10] where they can make further modifications to their playlist before sharing. If any changes were made to the Share Playlist Page [10] before clickBack, the changes are saved.

selectManualAdd
On the Share Playlist Page [10] the family member may select a family member who does not yet have an account registered in the system. If this is the case, the family member will be prompted for the family member’s email via the Prompt for Email pop-up.

enter(playlist, recieverEmail)
[FamilyMemberAccount.Email = receiverEmail &&
∄ (playlist.FamilyMemberAccount.Email = receiverEmail)]
/playlist.FamilyMemberAccount := FamilyMemberAccount;
sendNotification(FamilyMemberAccount)
Once prompted for the new family member’s email, the family member must enter the receiver’s email. This will display the Email Sent Confirmation pop-up.

ok
From the Email Sent Confirmation pop-up, the family member selects the “Ok” control and is brought back to the Share Playlist Page [10] to send more invites.

selectExisting(playlist, selectedEmail)
[∃ (playlist.FamilyAccount.Email = selectedEmail)]
/playlist.FamilyMemberAccount := FamilyMemberAccount;
sendNotification(FamilyMemberAccount)
On the Share Playlist Page [10] the family member may select an existing family member to share the playlist with. If this is the case, the family member will select the existing family member account(s) from the list on this page.

openPlaylistInvite
Once a playlist has been shared with another family member, they will receive a notification via our in-app notification service. Clicking on this playlist invite notification will bring the family member to the Playlist Page [11].

shareLater
/mode := “draft”
At any point on the Share Playlist Page [10], the family member may choose to share the playlist later. This transition will save the playlist as a draft and bring the family member back to the Playlist Page [11].

clickConfirm
From the Share Playlist Page, once the “Confirm” control is selected, the playlist is officially created and the family member is brought back to the Playlist Page [11].

play
/isPlaying := true
A media file is being played by the media player.

stop
/isPlaying := false
There are no media files being played by the media player.

Concurrent Machine (Transition Legend)

files := [ ],
isPlaying:= false,
selectedMediaFile := 0,
increment := false
Once the concurrent machine state is entered, there are no media files in the playlist, hence, the files[ ] array is initialized to empty. Moreover, there is no file playing currently so this variable is set to false and there is no media file selected, hence, the selectedMediaFile variable is initialized to 0. Increment is initialized to false.

when(files.empty())
If there are no media files in the playlist, that is, files[ ] is empty, then the family member must make a media file selection. Therefore, this transition moves to the Media Selection state. Moreover, if there are no files left in the selected playlist, the system transitions back to the Idle state.

when(selectedMediaFile > files.length)
/selectedMediaFile := 0
The selectedMediaFile variable indicates the index in the files[ ]. Therefore, when the selectedMediaFile variable is greater than the files[ ] array, this is invalid behavior and the variable is reset to zero. The family member must then select another media file to play, hence the transition returns to the Media Selection state.

after(files[selectedMediaFile].Duration)
/increment := true
Once a file that was playing finishes playing, the media player needs to move on to the next file as indicated by setting the increment variable to true.

when(isPlaying)
Once a media file starts to be played by the media player (when the isPlaying variable is true), the media player starts to play. As long as isPlaying is still true, the system will continue to switch between the Media Selection and Media Player Plays states to continuously queue up the next file to be played, and then play that file.

when(files)
/selectedMediaFile := 0
If there is at least one file in the playlist, that is, files are non-empty, then the selectedMediaFile variable is set to zero to point to the first file of the files[ ] array.

after(1)
[isPlaying == true]
The concurrent machine periodically assesses if a media file has been selected to play. If this is the case, then the isPlaying variable must be set to true. This transition then leads to the Media Player Plays state.

after(0)
[increment]
/selectedMediaFile := selectedMediaFile + 1,
increment := false
When the increment variable is true, the selectMediaFile variable increases by one to indicate that the media player is moving onto the next file in the list of files. This also sets the increment variable to false as the next file being played is not finished playing.

3.3 Quality Requirements

Quality requirements interviews were performed in order to obtain the top three quality attributes interviewees prioritize. The 100-dollar prioritization technique was used and its associated chart can be seen below:

	Scores					
Requirement	Young Adult A	Young Adult B	Young Adult C	Young Adult D	Busy Parent A	**Sum**
Security	10	20	22	20	20	**92**
Usability	25	25	19	30	25	**124**
Performance	50	20	19	30	25	**144**
Reliability	5	15	13	10	10	**53**
Robustness	5	10	16	5	15	**51**
Adaptability	5	10	11	5	5	**36**

Total Ordering

As a result of the 100-Dollar prioritization activity, we were able to elicit the total priority ordering of quality attributes. The results were (1) Performance, (2) Usability and (3) Security. This was the priority ordering for all stakeholders across the board. For all five stakeholders for which the 100-Dollar prioritization activity was performed, their top three quality attributes were performance, usability and security.

Relative Priority

Since it is clear that users overwhelmingly feel that performance, usability and security are important attributes, we can now discuss the relative priority ordering of these attributes. Young Adult A weighed the remaining three qualities, reliability, robustness and adaptability the lowest among all stakeholders with a value of $5 for each. Other stakeholders were not too far off from this mark with scores of about $10±5 for each of the remaining qualities. When it came to the top three quality attributes, Young Adult A placed the most value in the performance attribute, allocating $50. The remaining four stakeholders each allocated approximately $20-$30 to each to the performance, usability and security attributes respectively.

The reason why stakeholders may feel these qualities were not as important as performance, usability and security is because these qualities are less user-centric in comparison. For instance, adaptability may be a quality that is more business-focused as stakeholders may not have a great idea of how software maintainability works in relation to adjacent systems.

The following are descriptions of the top three quality attributes as well as the means in which these quality attributes are being measured.

Category: Security

- *Measurement:* Number of reports of accounts being breached per month.

It is important that family data, such as schedules, are stored securely on our servers and are private to the family members. To quantify this, the number of reported security breaches is a good start in determining the effectiveness of our security protocols. A low number of reports indicate these mechanisms are working appropriately while a high number of reports may indicate that more adequate security solutions need to be implemented.

Category: Usability

- *Measurement:* Percentage of users who find our system easy to use.

Since our target demographic ranges quite vastly in age, the tech-savviness of our users is not guaranteed. As a result, we need to be proactive in ensuring our interface is easy to use and not overcomplicate the lives of the families using our system. As a means of measuring this attribute, the percentage of users who feel our interface is easy to use is used.

Category: Performance

- *Measurement:* Time it takes to open a poll to plan a family activity on our platform (in seconds).

Tasks should take seconds to complete. If the execution time of these processes takes several minutes, this would strain the load on our system and result in poor user interactions. Given that our most important use case, according to Kano prioritizations, is planning a family activity, the best method of measuring this attribute would be the execution time of completing an action in this task's workflow in seconds. Interviewees were asked to estimate the execution time of opening a poll to plan a family activity.

From the quality requirements interviews, we were also able to deduce the rich fit criteria for the top three quality attributes. These criteria are summarized in the below table:

Requirement	Outstanding (Average)	Outstanding (Standard Deviation)	Target (Average)	Target (Standard Deviation)	Minimum (Average)	Minimum (Standard Deviation)
Security: Number of reports of accounts being breached per month	0	0	1.8	1.789	3.8	3.493
Usability: Percentage of users who find our system easy to use	98%	4.472%	84%	6.519%	71%	11.402%
Performance: Time it takes to open a poll	0.94s	0.627s	2s	1.732s	4.2s	1.643s

The above consolidated results highlight the average and standard deviation of all interviewees' rich fit criteria. Coincidently, all five stakeholders interviewed had the same priority ordering, with security, usability and performance being the top three attributes as per the 100-Dollar prioritization activity. Hence, there is no available data for reliability, robustness and adaptability attributes. Perhaps a larger sample size would more accurately depict the priorities of these latter three attributes.

Based on the standard deviations for performance and security measurements, our findings appear to be fairly accurate. The deviations for these attributes lay within an acceptable threshold (i.e. ±2s and ±1s for performance). This is because a standard deviation that is small indicates that the data lies close to the mean. Ideally, data that is within ±2 standard deviations is considered to be closer to the true value of that measurement. The standard deviations calculated for usability measurements appear to vary the most with the highest deviation being about ±11% from the mean. This could imply that further interviews are necessary to obtain more precise results.

References

[1] "API Reference," n.d. Accessed on: Dec. 9, 2021. [Online]. Available: https://developers.google.com/calendar/api/v3/reference

[2] M. Pandurov. *25+ Crazy Cell Phone Statistics for Canada [for 2021].* Reviewlution. n.d. Accessed on: Dec. 9, 2021. [Online]. Available: https://reviewlution.ca/resources/cell-phone-statistics-canada/.