AllEventlу: Design Document

By Benjamin Bruyns, Peggy Lewis, and Spenser Morey

Requirements go here…. 5
Design Description:. 6
UI Design:. 6
Login:. 6
Signup:. 6
Password Reset:. 6
Events Page:. 6
Public Events Page:. 7
Account:. 7
Event Creation:. 8
Event Chat:. 8
Guest List:. 8
Invitation:. 8
Server Design:. 9
Load Password Reset Page Button:. 9
Load Create Account Page Button:. 9
Load Google Login Page Button:. 9
Login Button:. 9
Sign-Up Button:. 9
Password Reset Button:. 9
Cancel Button:. 9
Update Password Button:. 10
Create Event Button:. 10
Load Login Page Button:. 10
Account Page Save Button:. 10
Account Page Reset Button:. 10
Event Chat Send Button:. 10
Invitation Accept Button:. 10
Invitation Decline Button:. 10
Invitation Login Button:. 11
Print Invitation Button:. 11
Navigation Menu User Name Hyperlink:. 11
Navigation Menu Event Button:. 11
Navigation Menu Public Events Button:. 11
Navigation Menu Sign-Out Button:. 11
Cancel Event Button:. 11
Guest List Button:. 11
Event Chat Load Button:. 11
Edit Button:. 12
View Invitation Button:. 12

Event Creation Save Button:....12
Send Invitations Button:....12
Database Design:....12
Login:....12
Signup:....12
Password Reset:....13
Update Password:....13
My Events:....13
Public Events:....13
Account:....13
Event Creation:....13
Event Chat:....14
Guest List:....15
Invitation:....15
Appendix....16
Message Documentation:....16
Sign-Up Button Message:....16
Event Chat Send Button Message:....17
Login Button Message:....17
Event Creation Message: //needs to be updated....18
Event Creation Save Button Message:....19
Load Create Account Page Button Message:....19
Load Google Login Page Button Message:....20
Load Password Reset Page Button Message:....20
Cancel Button Message:....20
Create Event Button Message:....20
Load Login Page Button Message:....21
Account Page Reset Button Message:....21
Guest List Button Message:....21
Event Chat Load Button Message:....22
Edit Button Message:....22
View Invitation Button Message:....22
Navigation Menu User Name Hyperlink Message:....22
Navigation Menu Event Button Message:....23
Navigation Menu Public Events Button Message:....23
Navigation Menu Sign-Out Button Message:....23
Print Invitation Button Message:....23
Invitation Login Button Message:....24
Update Password Button Message:....24
Password Reset Button Message:....24
Invitation Decline Button Message:....25

Cancel Event Button Message:.....26
Invitation Accept Button Message:.....26
Account Page Save Button Message:..... 27
Send Invitations Button Message:..... 27

Requirements:

See Requirements.

Design Description:

UI Design:

Login:

Both the Login page and the Sign-Up page feature a clean and minimalist design with a large image covering half of the page, setting a tone that is visually engaging for the user. On the Login page, an image is on the right, and on the left, the form fields for inputting email and password are positioned against a white background. This creates a strong contrast and directs the user's attention to the input fields. The Login page also includes a logo, a button to log in, a hyperlink to create an account, a hyperlink to reset a password, and an option to log in through Google's "personalized sign-in button." In case of an error, the Login Errors are red error messages to clearly indicate invalid input, ensuring users can easily identify and correct their mistakes.

Signup:

On the Sign-Up page, the image is on the left, with the form on the right side of the page. The Sign-Up page includes fields for inputting the user's first name, last name, email address, and password, with a second place to confirm the password. A sign-up button is in place to verify all the user's input and allow the user to create an account. The Sign-Up page also includes a logo, a cancel button that returns the user to the Login page, or an option to sign-up using Google's "personalized sign-up button." Suppose the user makes errors, such as mismatched passwords or invalid email formats. In that case, the Sign-Up Errors provides immediate feedback in red, prompting the user to correct the issues before submitting the form again.

Password Reset:

The Password Reset page is simple and functional, featuring an email input field where the user can request a password reset link. The user can submit their email using the reset password button or cancel this process using the cancel button that returns them to the Login page. Upon submitting their email, the user is guided through resetting their password. First, a dialog box will let the user know that if the email address is valid, a code has been sent to their email. This email will be used on the Update Password page, which also contains fields for inputting their new password and a field for confirming the password. This design is minimal to focus solely on resetting the password. Once all the fields are filled in correctly, the update password button will make the required changes, but if the user wishes to cancel, they can still opt to cancel this process by clicking the cancel button that returns them to the Login page.

Events Page:

After successfully completing login or sign-up, the user is directed to the Events page. This is the central hub where users can view the events they are attending and those they are hosting. There are filters for "Attending" and "Hosting" to display the desired events on a scrollable list. Each event is represented as a card with a clean design featuring key details, such as the event title, date and time, and location. It has a note if it is a public event or a private event.

For hosted events, the user's name appears as the host, along with a summary of the number of attendees. Quick actions like "Edit Invitation," "Guest List," “Print,” and “Event Chat” are accessible via buttons directly on the event card. For events the user is invited to or attending, the host’s name is shown along with a clear indication of whether the user has RSVP’d, with their guest count, if applicable. Guest quick actions include “View Invitation” and “Event Chat.”

To the left of the main content, a vertical sidebar is present, featuring simple and recognizable icons such as a person (indicating Account page), a group of people (linked to the Public Events page), a calendar for event management, and a sign-out button to return to the Login page. The interface prioritizes clarity, making it easy to track and manage multiple events at once.

Public Events Page:

The Public Events page in AllEvently provides users with a visually organized and dynamic way to explore upcoming public events. The layout maintains consistency with the rest of the platform, featuring the familiar "AllEvently" branding at the top left and a vertical sidebar with easily recognizable icons for quick navigation.

At the top of the main content area, the large, bold header "Public Events" clearly identifies the purpose of the page. Below this, a search bar allows users to filter through events by name, accompanied by a magnifying glass icon for intuitive interaction. Directly next to the search bar are calendar and filter icons, giving users quick access to refine events by month, week, or day.

The Popular Upcoming Events section is prominently displayed, showcasing a series of event cards in a horizontal, scrollable format. Each card contains key details about the event, including the event name, date, and time, as well as the location and host's name.

Each event card displays the invitation photo, helping to differentiate events visually. This design makes it easy for users to scan through the list and quickly identify events they are interested in attending. Overall, the Public Events page is clean, functional, and user-friendly, offering a seamless experience for browsing and engaging with public events on the platform.

Account:

The Account page provides users with an intuitive interface for managing their personal information. At the top, the "AllEvently" logo is prominently displayed, reinforcing the platform’s branding. Below the logo, a bold and large "Account Settings" header makes it clear to the user which section of the application they are currently interacting with. The sidebar retains the recognizable icons, offering quick navigation to other sections like the Account page, Public

Events page, back to the Events page, or sign-out. The main body of the page contains text fields labeled "First Name" and "Last Name," where users can modify their personal information. These fields are accompanied by placeholder text for the user's first and last name to give the user a visual of the current information. A "Reset Password" hyperlink is positioned below the name fields, offering users a quick way to initiate a password change via the Password Reset page. Below this, a "Save" button is clearly visible, giving the user a simple action to submit their changes.

The page layout is clean, and the interactive elements are spaced evenly, contributing to a user-friendly experience that ensures easy navigation and editing of account details.

Event Creation:

The Event Creation page is one of the most visually rich sections of the platform. The top features the standard "AllEvently" logo and the save button. A sidebar on the left contains an event name and preview pane, displaying a small section where users can input and preview event details in real time.

In the main body of the page, users are guided through a detailed form for creating and customizing events. Sections are divided with clear, bold headers such as "Event Name," "Note for Guests," and "Date and Location," helping users organize the information they input. Under "Date and Location," a toggle allows users to choose between a Single Event or a Recurring Event, with subsequent date and time fields that become enabled based on their selection.

A Google Map widget is integrated into the location section, allowing users to input and verify event addresses with real-time map feedback. This visual aid ensures that the event's location is clear to both the host and invited guests. Beneath the map, additional sections such as "Guest Settings" and "Host Settings" provide further options for managing guest RSVPs, child counts, and notifications, offering flexibility for event organizers.

The page is designed to handle complex input tasks without overwhelming the user, and the accordion-style sections collapse when not in use, keeping the interface clean and focused.

Event Chat:

The Chat Page in AllEvently provides a seamless and user-friendly interface for real-time communication between event hosts and guests. The page is designed to resemble modern chat applications, with clearly differentiated message bubbles and timestamps to organize the conversation.

At the top, the AllEvently logo maintains the platform's branding, and below it, the main chat area is presented. Each message is displayed in a speech-bubble format, with the sender's name and timestamp clearly visible at the top of each bubble. Messages sent by the user are shown in a bright blue bubble on the right side of the screen, helping to distinguish them from the other participants' messages, which are in grey on the left. The alternating colors make it easy for users to follow the flow of conversation.

For example, in this chat, the user asks for gift ideas for a birthday party, and the response from "Emily" is neatly displayed in a grey bubble. The chat includes timestamps for all messages, such as "09/11/2024 10:37 AM," ensuring that users can track the timing of each message.

At the bottom of the chat window, a text input box allows users to easily type and send messages. A camera icon on the left provides the ability to attach images, and a paper airplane icon on the right allows the user to send the message with a single click. This clean and intuitive interface ensures that event attendees can easily communicate with each other in real-time, fostering engagement and collaboration for event planning or coordination.

Guest List:

The Guest List page in AllEvently offers a clear and structured way for event hosts to manage their guest invitations and RSVPs. The page is divided into three distinct sections: (at the top) the New Guest form on the left and the Guest List display on the right, providing an intuitive interface for both adding new guests and tracking existing invitees.

On the left side, the New Guest form allows hosts to quickly add new attendees by entering their First Name, Last Name, and Email into the corresponding fields. The form is straightforward, with placeholders in each field to guide the user through the process. Above the form, the friendly prompt "Let's invite someone!" encourages the host to begin adding guests. Once the information is filled out, the host can click the prominent blue Add Guest button to send an invitation.

The right side of the page displays the Guest List, showing a summary of all invited guests. At the top, the guests are categorized into three groups—Attending, Waiting, and Regrets—with numerical indicators showing the count of guests in each category. This feature helps hosts quickly assess how many people have responded and how many are still pending. The search bar allows users to filter through the list of guests, making it easy to find specific individuals.

Each guest is represented by a row, which includes their name, status (e.g., "Sent" or "Not Sent"), and an envelope icon to indicate whether the invitation has been sent. Guests who have responded are sorted accordingly, and hosts can easily view details such as whether a guest has confirmed their attendance. The right-facing arrow at the end of each guest's row provides access to detailed information about that guest, where hosts can manage their RSVP status.

The second image reveals the Guest Information page for an individual guest. Here, the host can review details such as the First Name, Last Name, and Email of the guest. The invitation status, including the Invite Sent date and whether the guest has responded, is displayed at the top. Additionally, the page offers three main actions: sending an Email Invite, marking the guest as “Attending,” or marking them as “Declined.” Buttons for saving changes or deleting the guest are clearly visible at the top right, providing simple but powerful controls for managing the guest list.

Overall, the Guest List page provides a comprehensive view of all attendees, making it easy for event organizers to track RSVPs and manage communications with their guests. The layout is

clean and intuitive, ensuring that users can easily navigate and perform the necessary actions to coordinate their events effectively.

Invitation:

The Invitation Page in AllEvently presents a visually appealing and informative display for event invitations, ensuring that guests can easily view and respond to event details. The design centers around a clean, attractive layout chosen by the host, with a focus on the event's theme and vital information.

At the top of the page, the event title is prominently displayed in a customized font, immediately setting the tone of the event. Just below the title, the location details are provided with the venue name and full address, including a small map preview for easy reference. This map integrates with Google Maps, allowing guests to interact with the map for easy identification of the event's location.

The main image of the page chosen by the host distinguishes the theme of the event. This aesthetically pleasing visual adds a personal touch to the invitation and makes it more engaging for recipients.

Below the image, the event date and time are clearly displayed, ensuring that guests can see when the event is taking place at a glance. A current RSVP status is also shown, in this case, alerting the guest that they haven't yet responded to the invitation, marked in red for emphasis.

Guests are offered various interactive options, including opening the event chat to communicate with other attendees, reviewing the Host Information, or accessing a linked wish list via the My Wish List option. A personal note from the host follows, offering more detailed information about the event, including attire recommendations, expectations, or additional details to help guests prepare.

In the lower-right corner of the page, a large, easy-to-spot RSVP button is positioned, encouraging guests to confirm their attendance with just a single click. This streamlined process ensures that responding to invitations is both simple and visually inviting.

Overall, the Invitation Page effectively combines visual appeal with functionality, providing guests with all the necessary details and interactive elements in one well-organized, aesthetically pleasing format.

Invitation PDF:

The Printable PDF Invitation Page for AllEvently is designed to provide guests with a tangible and visually appealing invitation that can be easily printed or shared. The layout is structured to deliver all the essential event details in a clean and organized manner while maintaining the theme of the event.

On the left side of the page, the chosen image is displayed, which reinforces the celebratory nature of the event. Below this, there is a large QR code, making it easy for guests to RSVP. The instructions underneath the code explain the process: "Open the camera on your phone and scan the QR (barcode) to RSVP." This ensures that even guests unfamiliar with RSVP processes can quickly understand how to confirm their attendance.

On the right side, all event details are prominently displayed. At the top, the phrase "You're Invited To:" introduces the event, followed by the event name, in a decorative font that adds a personal touch. The host's name is mentioned directly below to personalize the invitation further.

The event date and time are shown with clear icons, ensuring that the information is easy to spot. The calendar icon highlights the date, and a clock icon next to it shows the event time, providing clear visual markers for when the event will take place.

At the bottom right, a map displays the event venue's location, Wilma Rudolph Event Center, along with the full address, making it easy for guests to identify where the event will be held. The map integrates key location markers, such as nearby landmarks, providing guests with a sense of familiarity when planning their route.

Overall, the Printable PDF Invitation is designed for clarity and convenience, ensuring that all the important information is easily accessible and that guests have a straightforward way to RSVP via the QR code. The balance of aesthetic elements, such as the decorative font and playful imagery, with practical features like the map and RSVP instructions, makes this invitation both functional and visually inviting.

Server Design:

Load Password Reset Page Button:

After being pressed, the password reset button will send a message to the server indicating that it needs to load the Password Reset page. The server will process this message and respond by loading the requested page.

Load Create Account Page Button:

After being pressed, the create account button will send a message to the server requesting the Create Account page to be loaded, and the server will respond by loading the requested page.

Load Google Login Page Button:

After being pressed, the create account button will send a message to the server requesting the Google Login page to be loaded, and the server will respond by loading the requested page.

Login Button:

After this button is pressed, it will send a message to the database containing the information entered by the user. The database will then respond with a validation message. If this message indicates that the login was valid, the server will load the Event page. If the login was invalid, the user will be displayed an error message from the UI.

Sign-Up Button:

After entering their information on the Sign-Up page, the user will press this button. After being pressed, it will send a message to the database including the user's information. The database will respond with a message indicating whether their sign-up was valid or not. If it is valid, the server will load the Event age. If the message indicates that the sign-up was not valid, the server will load a message indicating to the user that the sign-up failed.

Password Reset Button:

After this button is pressed, it will send a message to the database which will retrieve the email of the user who pressed the button. The database will then send a message to the server, and the server will send a password reset link to the user's email address. The user will then be displayed an "ok" button which will send a message to the server after being pressed indicating that the login page needs to be loaded, and the server will respond by loading the requested page.

Cancel Button:

After being pressed, the Cancel button will send a message to the server requesting the Login page to be loaded, and the server will respond by loading the requested page.

Update Password Button:

Upon being pressed, the server will retrieve the email address from the HTML, assign a new password to the corresponding email, and return a success message to the user to notify them that their password was successfully updated.

Create Event Button:

After being pressed, the Create Event button will send a message to the server requesting the Event Creation page to be loaded, and the server will respond by loading the requested page.

Load Login Page Button:

After being pressed, the create account button will send a message to the server requesting the Login page to be loaded, and the server will respond by loading the requested page.

Account Page Save Button:

After being pressed, the Account Page Save button will send a message to the database containing the user's updated information, and the database will respond by updating the user's information accordingly.

Account Page Reset Button:

After being pressed, the Account Page Reset button will send a message to the server requesting the Reset Password page to be loaded, and the server will respond by loading the requested page.

Event Chat Send Button:

After this button is pressed, the server will send a message to the database containing the information from the event chat message, and the database will add this message content to the message list in the appropriate table.

Invitation Accept Button:

After being pressed, the server will send a message to the database to indicate that the user accepted the invitation, and the database will add the user to the list of users who have accepted the invitation. The button will also send a message to the server requesting the User Events page to be loaded, and the server will respond by loading the requested page.

Invitation Decline Button:

After being pressed, the server will send a message to the database to indicate that the user declined the invitation, and the database will add the user to the list of users who have declined the invitation.

Invitation Login Button:

After being pressed, the Invitation Login button will send a message to the server requesting the Login page to be loaded, and the server will respond by loading the requested page. The server will then send a message to the database, and the database will respond by adding the event to the list of events the user is attending.

Print Invitation Button:

After being pressed, the button will send a request to the server, and the server will respond by initiating a download of the PDF version of the invitation.

Navigation Menu User Name Hyperlink:

After being pressed, the hyperlink will send a message to the server requesting the Account page to be loaded, and the server will respond by loading the requested page.

Navigation Menu Event Button:

After being pressed, the button will send a message to the server requesting the User Event page to be loaded, and the server will respond by loading the requested page.

Navigation Menu Public Events Button:

After being pressed, the button will send a message to the server requesting the Public Events page to be loaded, and the server will respond by loading the requested page.

Navigation Menu Sign-Out Button:

After being pressed, the button will send a message to the server requesting the Login page to be loaded, and the server will respond by loading the requested page.

Cancel Event Button:

After being pressed, the button will send a message to the server. If the event is private, the server will process the message and send email notifications to all guests on the invite list. If the event is public, the server will send email notifications to all guests in attendance.

Guest List Button:

After being pressed, the button will send a message to the server requesting the Guest List page to be loaded, and the server will respond by loading the requested page.

Event Chat Load Button:

After being pressed, the button will send a message to the server requesting the Event Chat page to be loaded, and the server will respond by loading the requested page.

Edit Button:

After being pressed, the button will send a message to the server requesting the Event Creation page to be loaded, and the server will respond by loading the requested page.

View Invitation Button:

After being pressed, the button will send a message to the server requesting the Invitation page to be loaded, and the server will respond by loading the requested page.

Event Creation Save Button:

After being pressed, the server will send a message to the database containing the information provided by the user, and the database will return with a validation message depending on whether the user entered the necessary information to save.

Send Invitations Button:

After being pressed, the button will send a message to the server requesting the Guest List page to be loaded, and the server will respond by loading the requested page.

Database Design:

Login:

To verify if a set of credentials is valid, the database requires the email and the encrypted password of the account accessed from the server. The database will check the Accounts table for an account that matches the credentials. If not, the database will return NULL. If so, a session will be generated for that account in the Sessions table. The session ID will then be returned to the server to permit the user access.

Signup:

To create an account, the database requires the email, password, first name, and last name provided by the server. The database will check the Accounts table for an entry with that email address. If so, then the database will return NULL. If not, the database will create an entry in the Accounts table. Then the database will check if there is an entry in the People table that matches the email address provided by the server. If not, one will be created using the email, first name, and last name provided by the server. If so, the first and last names of the matching entry will be updated to show the first and last names provided by the server. If a new account is created, a session will be generated for that account in the Sessions table. The session ID will then be returned to the server to permit the user access.

Password Reset:

To begin a password reset, the database requires the account's email address from the server. The database will search the Accounts table for an account with that email address. If there is no match, the database will return NULL. If there is a match, the database will create an entry in the Reset_Credentials table with the provided email and a randomly generated password. The generated password will then be returned to the server. The entry in the Reset_Credentials table will exist for ten minutes and then be deleted.

Update Password:

To update a password for an account, the database requires an email address, a temporary password, and a new password. The database checks the Reset_Credentials table for an entry that matches the given email address and temporary password. If a match is found, the database updates the entry in the Accounts table that contains the email given such that the given new password is now the password of that entry.

My Events:

To provide a list of events a given user is the host of, the database requires that user's email from the server. Then the database will search the Events table for events where the given email matches the host email of the event. The events found this way are returned to the server in descending order by start date.

To provide a list of events a given user is attending, the database requires that user's email from the server. The database will then search the Guests table for the event IDs of events the user of the given email address is attending. Then the database will search the Events table for events where the event ID matches an event ID of an event the user of the given email address is attending.

Public Events:

To provide a list of public events, the database requires no information. The database will simply search the Events table for public events. The events found this way are returned to the server in descending order by start date.

Account:

To change a user's first and last name, the database requires the email address of the user and the new first and last names from the server. The database will search for the People entry corresponding to the provided email. That entry's first and last names will then be updated to the new first and last names.

Event Creation:

To create a new event, the database requires the information for the event table except the event ID. Nullable values can be null. Using this information, the database will create a new entry in the Event with a unique event ID. The event ID will then be returned to the server.

To find a specific event, the database requires the event ID provided by the server. The database then searches the Events table to find the corresponding event. The information for the event is then returned to the server.

To store image information, the database requires the image and the email of the user uploading the image. Both provided by the server. The database will create an entry in the Images table using the image provided by the server a unique image ID generated by the database. The database will then create an entry in the User Images table using the email provided the serve and the unique image ID.

To retrieve all images that a user can access, the database requires an email address provided by the server. The database will search through the Images table for all entries where there is no corresponding entry in the User Images table with an email other than the specified email. All entries found in this way are returned to the server.

To find a specific image, the database requires the image ID. The database then searches the Images table for the corresponding entry. That entry is then returned to the server.

Event Chat:

To get all messages for an events chat, the database requires the event ID to be provided by the server. The database will search the Chat_Messages table for all entries with the provided event ID. All chat messages found this way are returned to the server in descending order by date.

To add a message to the chat, the database requires the sender's email address, the chat's event ID, the message's content, and whether it is an RSVP response from a guest (can be null). This information will come from the server. The database will then create a new entry in the Chat_Message table using the information provided, a unique message ID generated by the database, and the current date. The database then searches the Chat_Message table for entries that match the event ID provided by the server. Entries found this way are returned to the server in descending order by date.

Guest List:

To produce the guest list for an event, the database requires the event ID from the server. The database will then search the Guests table for guests attending the corresponding event. The database will join the names of the entries corresponding to the guests in the People table to the guest data. The results are then returned to the server.

To add a guest from the guest list with no response, the database requires the event ID, email of the guest, first name and last name of the guest provided by the server. The database will then search the People table for an entry with a matching email. If such an entry does not exist, the database will create one using the provided email address and first and last names. After searching the People table, the database will search the Guests table for an entry that matches both the event ID and the email provided by the server. If there is no match, the database will create an entry in the Guests table using the event ID email of the guest. The responded and sent fields will be "false." The accepted, child count, and guest count fields will be set to null. The database will then return the email to the server. If there is a match, the database will return null to the server.

To update that a guest has been sent the invite, the database requires the guest's email and the event ID. The database will find the corresponding entry in the Guests table and update the sent field to "true" for that entry. Then the database will return entries in the Guests table that correspond to the given event ID and return them to the server.

Invitation:

To determine if a guest is on the guest list of an event, the database requires the event ID and an email address provided by the server. The database then searches the Guests table to find

the match for the event ID and the email. If there is a match, the database returns true to the server. If not, the database returns false.

To add an RSVP when logged in, the database requires the event ID, email, whether the guest accepted, a child count (can be null), and a guest count (can be null) provided by the server. The database will then search the Guests table to find an entry corresponding to the given event ID and email. If one is found, the entry is updated using the information provided, and the responded field is updated to “true.” If no entry is found, the database creates an entry based on the given data and sets the responded field to true.

To add an RSVP when not logged in, the database requires the event ID, email, first and last name, whether the guest accepted, a guest count, and a child count (can be null). All are provided by the server. The database will search the Accounts table for an account with the given email. If there is no match, the database will add or update the entry in the People table corresponding to the given email with the provided first and last names. After searching the Accounts table, the database will search the Guests table to find an entry corresponding to the given event ID and email. If one is found, the entry is updated using the information provided, and the responded field is updated to “true.” If no entry is found, the database creates an entry based on the given data and sets the responded field to true.

Appendix

Block Diagram:

See BlockDiagram.

Component Diagram:

See AllEventlyClassDiagram.

Storyboard:

See AllEventlyUIStoryboard.

Message Documentation:

Sign-Up Button Message:

Purpose:
When pressed, the button sends a request to the database to register the user with their provided information and handles the response based on the validity of the sign-up.

Button Action:
Action: Submits user information for sign-up and loads the appropriate page based on the response
Method: POST
Endpoint: /sign-up

Request Body:
{

“firstName”: “string”,
“lastName”: “string”,
“emailAddress”: “string”,
“password”: “string”,
“confirmedPassword”: “string”
}

Response:
{
“valid”: “boolean”,
“message”: "Your sign-up was successful." | "Sign-up failed. Please try again."
}

Event Chat Send Button Message:

Purpose:
When pressed, the button sends a request to the database to save the event chat message provided by the user.

Button Action:
Action: Submits the event chat message to the database
Method: POST
Endpoint: /send-event-chat

Request Body:
{

“eventID”: “integer”,
“email”: “string”,
“content”: “string”,
“date”: “date”,
“accepted”: “boolean”
}

Response:
{
“message”: "Your message has been successfully sent."
}

Login Button Message:

Purpose:
When pressed, the button sends a request to the database to validate the user's login credentials and handles the response based on the validity of the login.

Button Action:
Action: Submits login information and receives a validation message
Method: POST
Endpoint: /login

Request Body:
{

“emailAddress”: “string”,
“password”: “string”
}

Response:
{
“Valid”: “boolean”,
“message”: "Login successful." | "Invalid login credentials. Please try again."
}

Event Creation Message: //needs to be updated

{
“eventHost”: “string”,

```
	“hostName”: “string”,
	“eventName”: “string”,
	“eventStartDate”: “date”,
	“eventEndDate”: “date”,
	“eventEndTime”: “date”,
	“eventTmeZone”: “string”,
	“isRecurring”: “boolean”,
	“eventFrequency”: “integer”,
	“endRecurrenceDate”: “date”,
	“eventLocation”: “string”,
	“requestChildCount”: “boolean”,
	“childCount”: “integer”,
	“limitAdditionalGuests”: “boolean”,
	“additionalGuestCount”: “integer”,
	“hostFirstName”: “string”,
	“hostLastName”: “string”,
	“hostNotifications”: “boolean”,
	“requireRSVP”: “boolean”,
	“isPublic”: “boolean”,
	“invitationLayout”: “string”,
	“backgroundImage”: “int”,
	“fontBackgroundColor”: “string”,
	“fontColor”: “string”,
	“font”: “string”
}
```


Event Creation Save Button Message:

Purpose:
When pressed, the button sends a request to the database to save the event information provided by the user and handles the response based on the validation of the entered information.

Button Action:
Action: Submits event information for creation and receives a validation message
Method: POST
Endpoint: /create-event

Request Body:


```
{
	“eventHost”: “string”,
	“hostName”: “string”,
```

“eventName”: “string”,
“eventLocation”: “string”,
“eventStartDate”: “date”,
“eventTmeZone”: “string”,
“invitationLayout”: “string”,
“backgroundImage”: “int”,
“fontBackgroundColor”: “string”,
“fontColor”: “string”,
“font”: “string”
“isPublic”: “boolean”,
“isRecurring”: “boolean”,
“requestChildCount”: “boolean”,
“limitAdditionalGuests”: “boolean”,
“hostNotifications”: “boolean”,
}

Response:
{
“Valid”: “boolean”,
“message” : "Event creation saved successfully." | "Event creation failed. Please provide all necessary information."
}

Load Create Account Page Button Message:

Purpose:
When pressed, the button sends a request to the server to load the "Create Account" page.

Button Action:
Action: Requests the Create Account Page
Method: GET
Endpoint: /create-account

Load Google Login Page Button Message:

Purpose:
When pressed, the button sends a request to the server to load the "Google Login" page.

Button Action:
Action: Requests the Google Login Page
Method: GET
Endpoint: /google-login

Load Password Reset Page Button Message:

Purpose:
When pressed, the button sends a request to the server to load the "Password Reset" page.

Button Action:
Action: Requests the Password Reset Page
Method: GET
Endpoint: /password-reset

Cancel Button Message:

Purpose:
When pressed, the button sends a request to the server to load the "Login" page.

Button Action:
Action: Requests the Login Page
Method: GET
Endpoint: /login

Create Event Button Message:

Purpose:
When pressed, the button sends a request to the server to load the "Event Creation" page.

Button Action:
Action: Requests the Event Creation Page
Method: GET
Endpoint: /create-event

Load Login Page Button Message:

Purpose:
When pressed, the button sends a request to the server to load the "Login" page.

Button Action:
Action: Requests the Login Page
Method: GET
Endpoint: /login

Account Page Reset Button Message:

Purpose:
When pressed, the button sends a request to the server to load the "Reset Password" page.

Button Action:
Action: Requests the Reset Password Page
Method: GET
Endpoint: /reset-password

Guest List Button Message:

Purpose:
When pressed, the button sends a request to the server to load the "Guest List" page.

Button Action:
Action: Requests the Guest List Page
Method: GET
Endpoint: /guest-list

Event Chat Load Button Message:

Purpose:
When pressed, the button sends a request to the server to load the "Event Chat" page.

Button Action:
Action: Requests the Event Chat Page
Method: GET
Endpoint: /event-chat

Edit Button Message:

Purpose:
When pressed, the button sends a request to the server to load the "Event Creation" page.

Button Action:
Action: Requests the Event Creation Page

Method: GET
Endpoint: /edit-event

View Invitation Button Message:

Purpose:
When pressed, the button sends a request to the server to load the "Invitation" page.

Button Action:
Action: Requests the Invitation Page
Method: GET
Endpoint: /invitation

Navigation Menu User Name Hyperlink Message:

Purpose:
When pressed, the hyperlink sends a request to the server to load the "Account" page.

Button Action:
Action: Requests the Account Page
Method: GET
Endpoint: /account

Navigation Menu Event Button Message:

Purpose:
When pressed, the button sends a request to the server to load the "User Event" page.

Button Action:
Action: Requests the User Event Page
Method: GET
Endpoint: /user-event

Navigation Menu Public Events Button Message:

Purpose:
When pressed, the button sends a request to the server to load the "Public Events" page.

Button Action:
Action: Requests the Public Events Page

Method: GET
Endpoint: /public-events

Navigation Menu Sign-Out Button Message:

Purpose:
When pressed, the button sends a request to the server to load the "Login" page.

Button Action:
Action: Requests the Login Page
Method: GET
Endpoint: /login

Print Invitation Button Message:

Purpose:
When pressed, the button sends a request to the server to initiate a download of the PDF version of the invitation.

Button Action:
Action: Requests PDF download of the invitation
Method: GET
Endpoint: /download-invitation

Invitation Login Button Message:

Purpose:
When pressed, the button sends a request to the server to load the "Login" page and add the event to the list of events the user is attending.

Button Action:
Action: Requests the Login Page and adds the event to the user's attendance list
Method: POST
Endpoint: /login-invitation

Update Password Button Message:

Purpose:

When pressed, the button sends a request to the server to update the user's password associated with their email address.

Button Action:
Action: Updates the password for the user's email
Method: POST
Endpoint: /update-password

Request Body:

```
{
	“email”: “string”,
	“newPassword”: “string”
}
```

Response:

```
{
	“message”: “Your password has been successfully updated.”
}
```

Password Reset Button Message:

Purpose:
When pressed, the button sends a request to the database to retrieve the email of the user and send a password reset link to that email address.

Button Action:
Action: Requests a password reset link and loads the login page
Method: POST
Endpoint: /password-reset

Request Body:

```
{
	"email": "string"
 }
```

Response:

```
{
	"message": "A password reset link has been sent to your email."
}
```

Subsequent Action:
After receiving the reset link, the user will be shown an “OK” button.

OK Button Action:
Action: Requests to load Login page
Method: GET
Endpoint: /login

Invitation Decline Button Message:

Purpose:
When pressed, the button sends a request to the server to indicate that the user has declined the invitation.

Button Action:
Action: Declines the invitation for the user
Method: POST
Endpoint: /decline-invitation

Request Body:
{
	“eventID”: “integer”,
	“userEmail”: “string”
}

Response:
{
	“message”: “You have successfully declined the invitation.”
}

Cancel Event Button Message:

Purpose:
When pressed, the button sends a request to the server to cancel the event and notify guests based on the event's visibility.

Button Action:
Action: Cancels the event and sends notifications to guests
Method: POST
Endpoint: /cancel-event

Request Body:
{
	“eventID”: “integer”,
	“isPublic”: “boolean”

}

Response:
{
	“message”: “The event has been canceled and notifications have been sent to guests.”
}

Invitation Accept Button Message:

Purpose:
When pressed, the button sends a request to the server to indicate that the user has accepted the invitation and to load the User Events page.

Button Action:
Action: Accepts the invitation and requests the User Events page
Method: POST
Endpoint: /accept-invitation

Request Body:
{
	“eventID”: “integer”,
	“userEmail”: “string”
}

Response:
{
	“message”: "You have successfully accepted the invitation."
}

Account Page Save Button Message:

Purpose:
When pressed, the button sends a request to the database to save the user's updated information.

Button Action:
Action: Updates the user's information in the database
Method: POST
Endpoint: /save-account

Request Body:
{
	 "userId": "string",

```
        "updatedInfo": {
                "firstName": "string",
                “lastName”: “string”,
        }
}
```

Response:

```
{
        “message”: "Your account information has been successfully updated."
}
```

Send Invitations Button Message:

Purpose:
When pressed, the button sends a request to the server to load the Guest List page for sending invitations.

Button Action:
Action: Requests the Guest List page
Method: GET
Endpoint: /guest-list

Storage Documentation

ERD:

For the database diagram, see AllEventlyERD.

Background Images:

Background image files will be stored as .PNG files on the server. These will be used to store images to recreate invitation images and display public events.