Product Requirements Document

PROJECT TITLE: CorgKey | **TEAM:** Fat Stacks
SPONSOR: Allthenticate | **MENTOR:** Bernie Conrad
MEMBERS: Adam Yu (Lead), Arjun Singh, Hunter Massey (Scribe), Olivia Gillam, Simon Yu

BACKGROUND:

Traditional security products (keys, smart cards, passwords, fobs, etc.) are inconvenient for users and are associated with a number of security risks. Without proper security infrastructure, people deal with memorizing passwords and carrying around keys and smart cards wherever they go, increasing the risk of getting locked out of their computers, accounts, servers, workplace, and even automobiles. Malicious actors can use these vulnerabilities to execute phishing attacks, stealing users' credentials and gaining access to their information and accounts. The risk posed by these attacks necessitates the use of additional security measures, including two-factor and biometric authentication.

Two-factor authentication is widely used online as another security measure in case one's password becomes compromised. Many companies have their own versions of 2FA in conjunction with either mobile phones (SMS or app) or with a separate email address. 2FA via SMS is vulnerable to SIM swapping attacks, making it less secure when compared to 2FA apps, such as Google Authenticator or Duo. Two-factor authentication still necessitates the use of a password, meaning that if a user forgets their password they must reset their password (usually via email). If given access to a user's email, hackers can exploit this vulnerability and change a user's password. This effectively bypasses 2FA, without relieving the user's burden of password management.

By creating an application to serve as the *primary* factor of authentication, users will not have to worry about managing passwords, nor will they be inconvenienced with 2FA apps or SMS codes. The application effectively replaces a user's password, eliminating the risk of phishing and allowing fast & convenient account access as long as a user carries their phone on their person. Users will also have easy access to

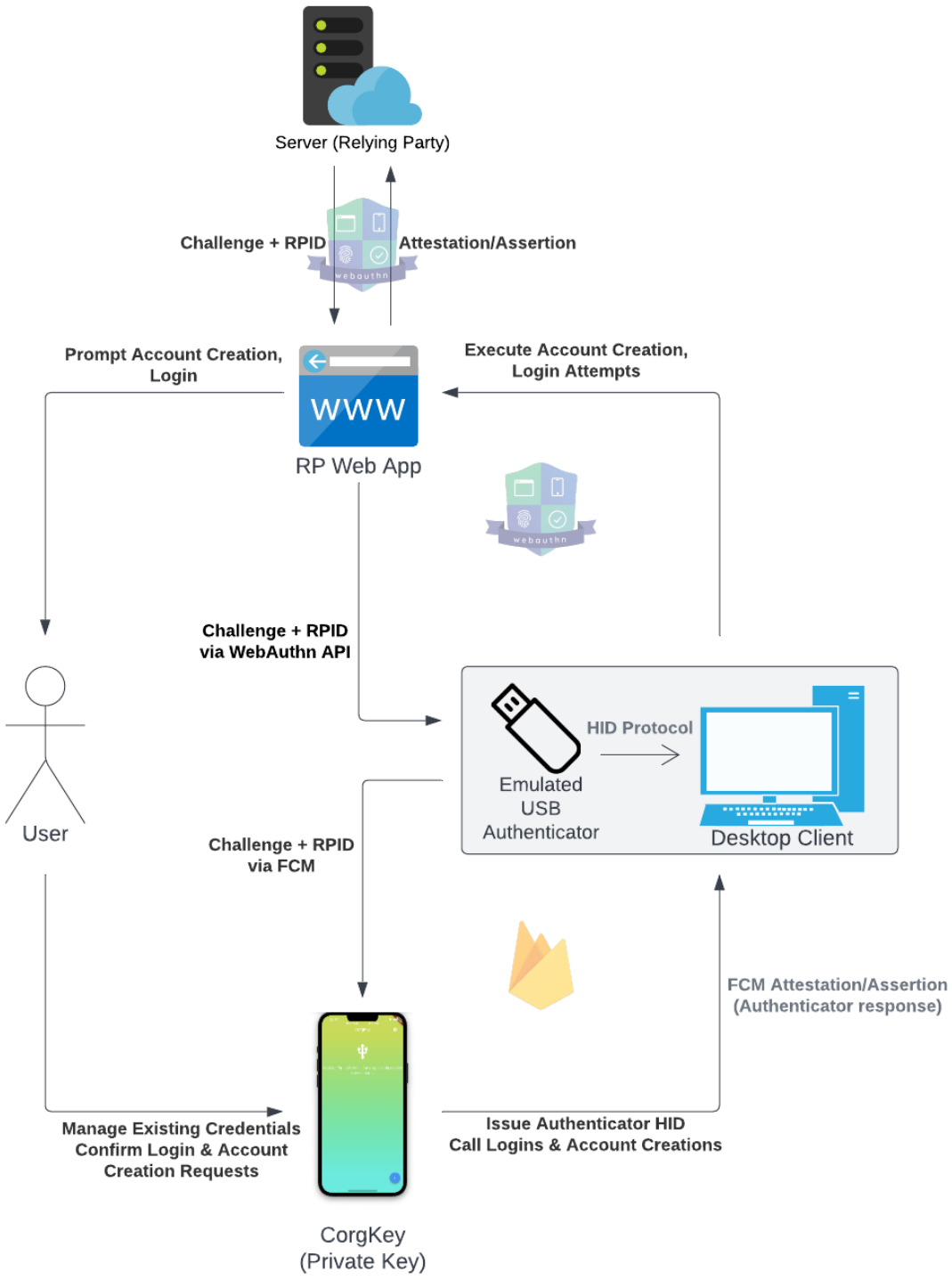
account management, allowing them to retain authorization to their accounts in the event that their phone is lost.

PROJECT GOALS:

With this project, we will create a mobile application that acts as a mobile roaming FIDO authenticator. Upon their first visit to a website on their computer, users will be prompted to create an account on their mobile application with minimal interaction and won’t need to come up with their own credential Upon future visits, users will be able to log into the website by authorizing the attempt on their phone, rather than manually entering their username and password.

SYSTEM ARCHITECTURE OVERVIEW:

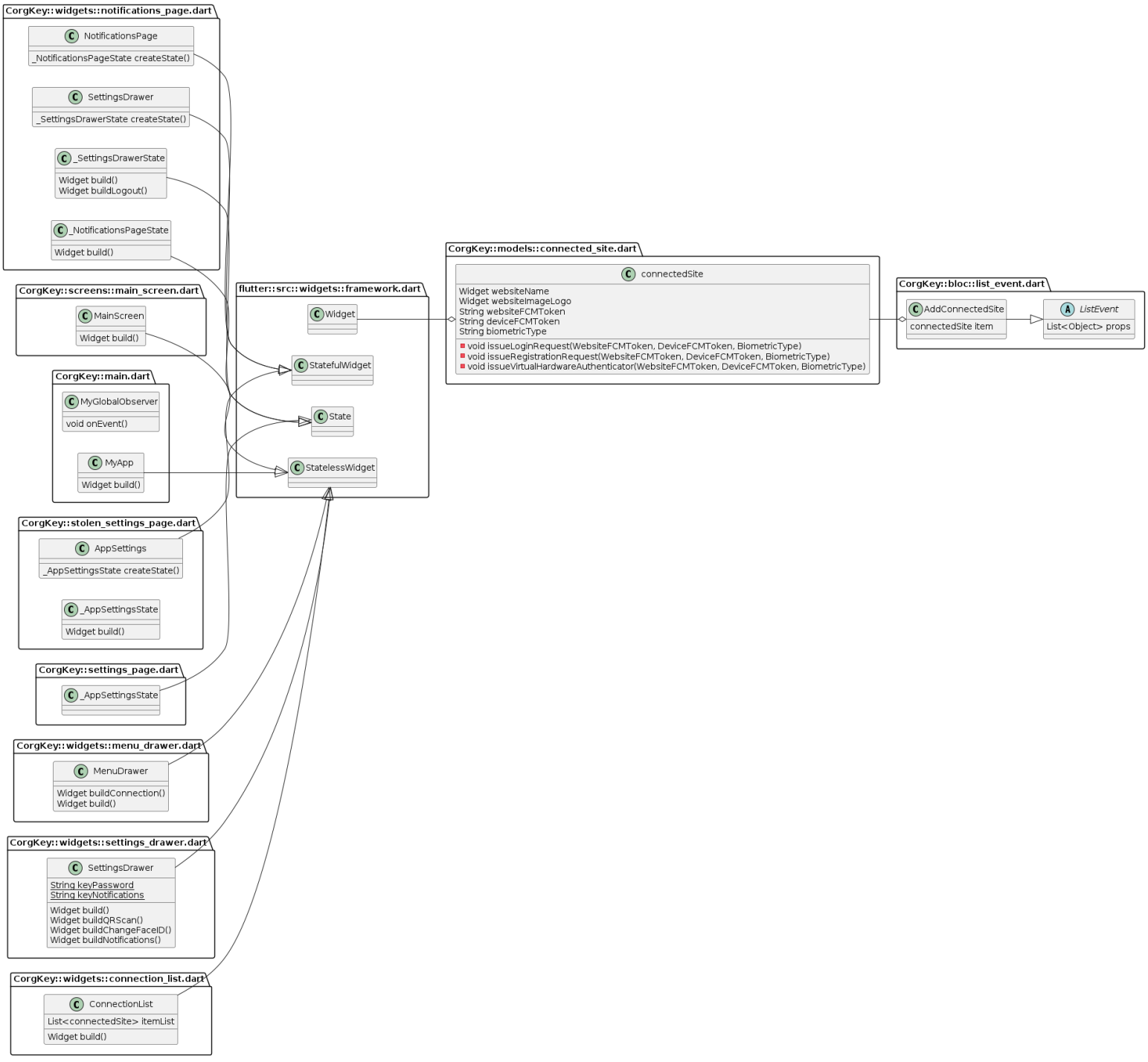
➔ High Level Diagram



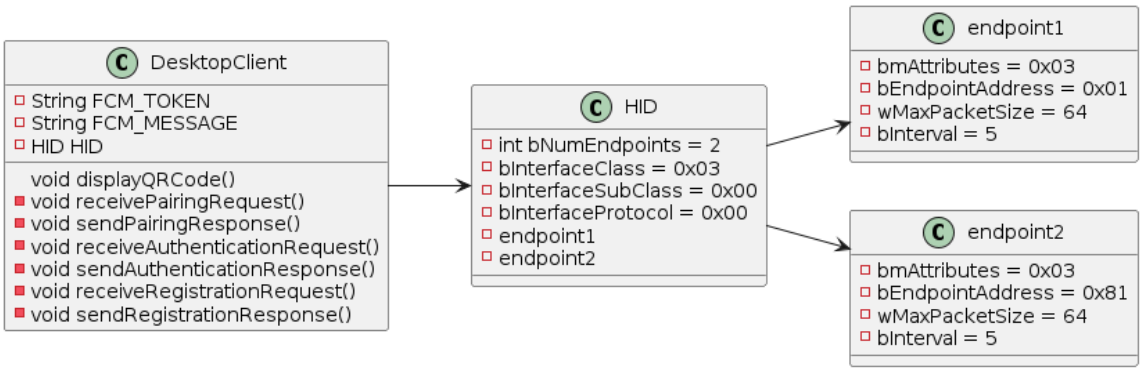
The diagram above shows a high-level architecture of our web application. The front end consists of the mobile application interface, along with the desktop client that will be used for device pairing and USB Human Interface Device (HID) emulation. Device pairing will tie a mobile device to a desktop via QR code, allowing the user to create accounts and login to web applications on that desktop using the instance of CorgKey installed on the mobile device.

Once CorgKey is paired with a desktop client, the client will use Python to emulate a USB HID with the properties of an authenticator device. The emulated HID authenticator will act as the FIDO authenticator device when account registration and login attempts are issued. Communication between the CorgKey mobile application and desktop will be done via Firebase Cloud Messaging (FCM). After receiving FCM commands from the mobile application, the desktop client will use the Client to Authenticator Protocol (CTAP) to communicate with the browser. The CTAP protocol will authorize account creation and login attempts through use of the emulated hardware authenticator.

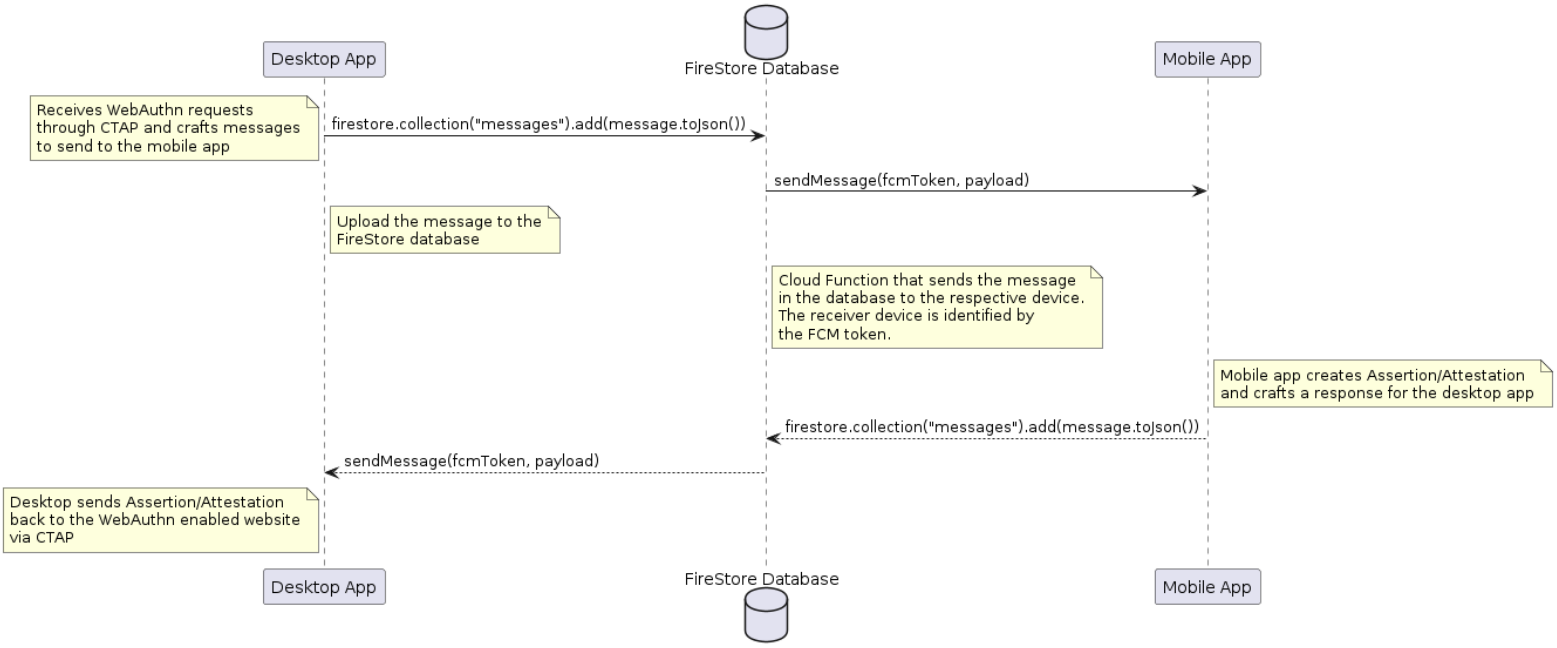
➔ UML Class Diagram for CorgKey App



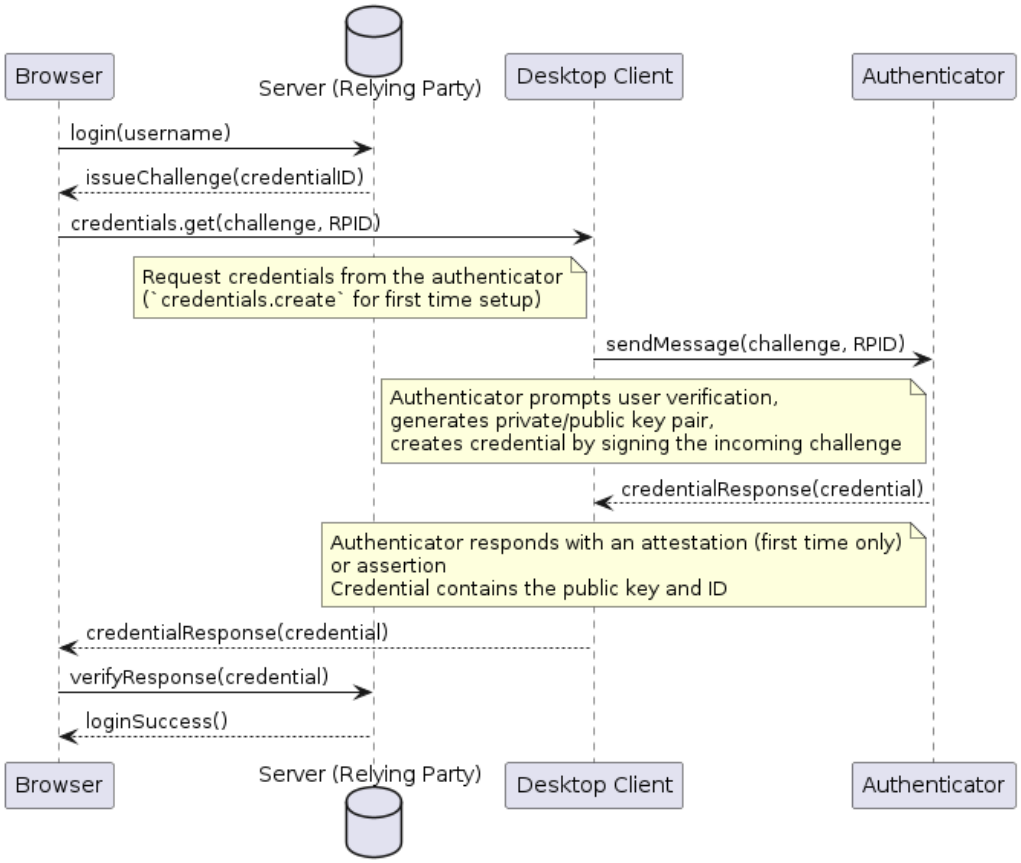
➔ UML Class Diagram for Desktop Client



➔ Sequence Diagram for Communication Between Desktop & Web



➔ Sequence Diagram for Authentication Process



USER INTERACTION & DESIGN

- **User Stories (& Associated Program Stories):**
  - User can create a new account on some site
    - User should see a request detailing where an account creation request has been issued from
    - CorgKey should send prompt to user asking for confirmation that the user wants to register with CorgKey
    - CorgKey should create new account associated with CorgKey identity and be able to log in in the future
    - Time estimate: 20 hours
    - GitLab Issue: https://gitlab.com/allthenticate-external/fido/app/-/issues/13

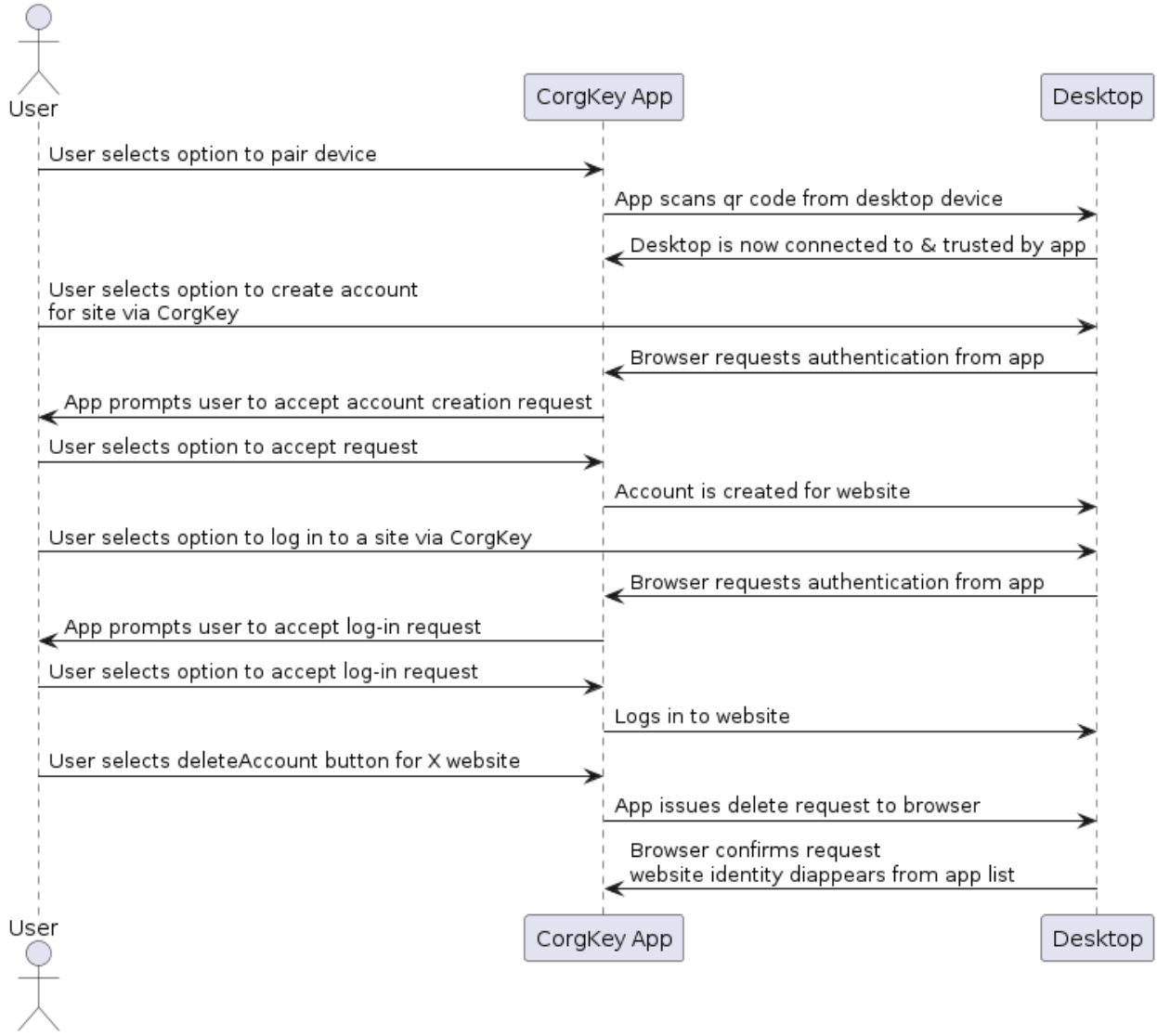
- User logs into account on some site
  - Websites should show option to log in with CorgKey, and then our app should process the request (if phone is connected to desktop client)
  - CorgKey should receive request from device that is trying to log in to account
  - CorgKey should send prompt to user asking for confirmation
  - CorgKey should send request back to site, allowing log in
  - Time estimate: 20 hours
  - GitLab Issue: https://gitlab.com/allthenticate-external/fido/app/-/issues/14
- User can view all sites that their authentication is connected to
  - CorgKey should have a scrollable alphabetical view of all site names, login info, and link to the login page
  - Time estimate: 2 hours
  - GitLab Issue: https://gitlab.com/allthenticate-external/fido/app/-/issues/15
- User can delete an account associated with a website
  - CorgKey should ask for confirmation
  - CorgKey should send request to website asking for account deletion
  - Time estimate: 2 hours
  - GitLab Issue: https://gitlab.com/allthenticate-external/fido/app/-/issues/16
- User can disable their account if their phone is lost
  - CorgKey should keep list of trusted external device(s); user can choose how many are required to disable, and can add/remove with verification
  - CorgKey should ask series of previously set up personal questions
  - Time estimate: 2 hours

        - GitLab Issue: https://gitlab.com/allthenticate-external/fido/app/-/issues/17
    - ~~User creates an account / User can log in~~
        - ~~Users can easily create an account and log in using email and password or some other method (e.g. Google OAuth)~~
        - ~~Program stores CorgKey account information in a database~~
        - ~~GitLab Issue: https://gitlab.com/allthenticate-external/fido/app/-/issues/12~~
        - ~~Time estimate: 3 hours~~
        - This issue has been deprecated; we prefer to manage access to CorgKey via a biometric or PIN factor, rather than a traditional username & password login.
    - User can set up a biometric or PIN factor
        - Upon device connection, user is prompted to set up a biometric or PIN factor (e.g. Apple FaceID)
        - If user sets up biometric/PIN, user will have to present the biometric or PIN factor to authorize login attempts on a device
        - Time estimate: 1 hour
        - GitLab Issue: https://gitlab.com/allthenticate-external/fido/app/-/issues/36
    - User can pair their mobile device with desktop client
        - Upon loading the desktop client, user is shown a QR code
        - User can use CorgKey to scan the QR code and pair their device with the desktop client
        - Time estimate: 8 hours
        - GitLab Issue: https://gitlab.com/allthenticate-external/fido/app/-/issues/37
    - User can send account creation from device to desktop client via FCM
        - Once the device is paired with the desktop client, users can confirm an account creation request on their device. This request will be communicated to the desktop client via FCM.

        - Time estimate: 3 hours
        - GitLab Issue: https://gitlab.com/allthenticate-external/fido/app/-/issues/38
    - User can send login request from device to desktop client via FCM
        - Once the device is paired with the desktop client, users can confirm a login request on their device. This request will be communicated to the desktop client via FCM.
        - This request can only be sent through if the user has already issued and confirmed an account creation request associated with the proper relying property.
        - Time estimate: 3 hours
        - GitLab Issue: https://gitlab.com/allthenticate-external/fido/app/-/issues/39
    - User is notified of successful emulation of USB authenticator device
        - Upon device being paired with desktop client, users are notified that a USB authenticator device has been successfully emulated
        - USB authenticator is emulated within the desktop client as an HID device
        - Emulated device must have the proper properties of an authenticator device
        - Time estimate: 12 hours
        - GitLab Issue: https://gitlab.com/allthenticate-external/fido/app/-/issues/40
    - User's list of connected sites must be persistent between instances of launching CorgKey
        - User should be able to close instances of CorgKey without losing their list of connected sites
        - List of connected sites should be persistent after closing & reopening CorgKey, rebooting phone, etc.
        - Time estimate: 2 hours

        - GitLab Issue: https://gitlab.com/allthenticate-external/fido/app/-/issues/41
    - User can use the desktop client to unpair their mobile device from the client
        - User should be able to access a dropdown menu in the client and choose to disconnect their mobile device
        - This action should sever the connection between CorgKey and the desktop client, along with deleting the emulated authenticator device
        - Time estimate: 3 hours
        - GitLab Issue: https://gitlab.com/allthenticate-external/fido/app/-/issues/42
    - User can opt into login notifications
        - User can access a dropdown menu in CorgKey for each connected site and choose to opt into login notifications for that site
        - If user opts in, they should receive notifications on the mobile device when the credential stored on the app is used to login
        - Time estimate: 2 hours
        - GitLab Issue: https://gitlab.com/allthenticate-external/fido/app/-/issues/43

USER INTERACTION SEQUENCE DIAGRAM

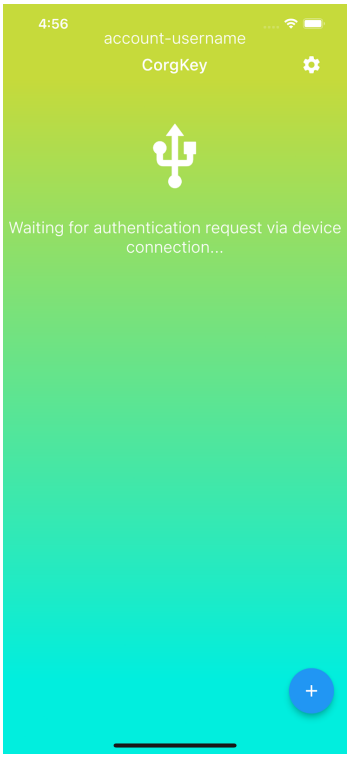


NON-FUNCTIONAL REQUIREMENTS:

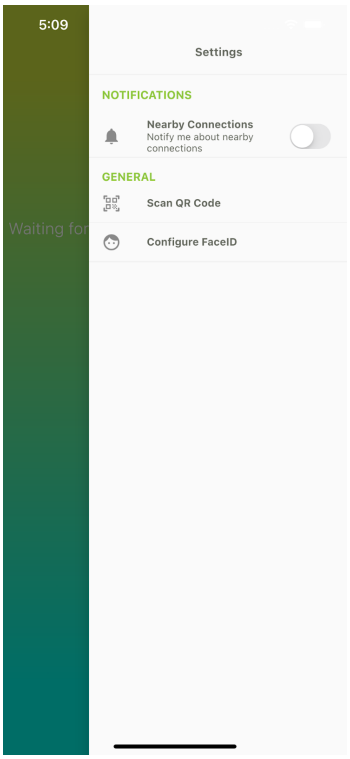
- The app should be intuitive to use
- The user should not be inconvenienced with frequently opening the app for authentication (e.g. Having the app run in the background)
- The authentication should be secure and fast
- There should be reliable methods for account recovery in the case of lost phone
- The application should be scalable to allow for many accounts

UI MOCKUPS

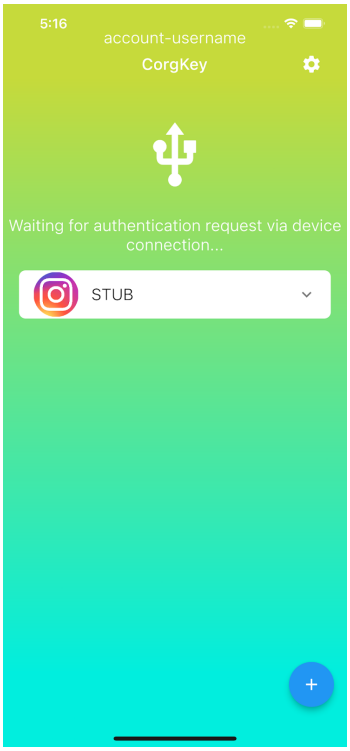
CorgKey Home Screen



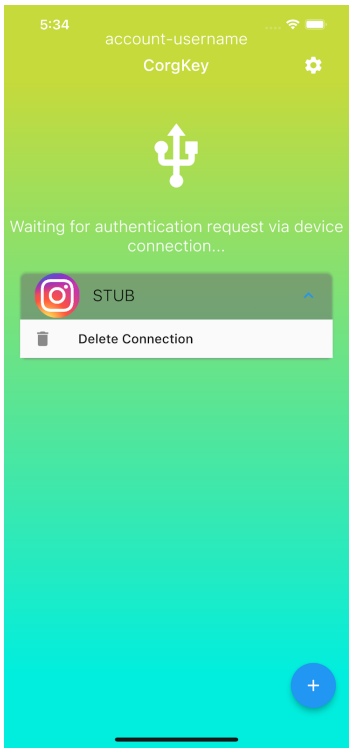
CorgKey Settings Drawer



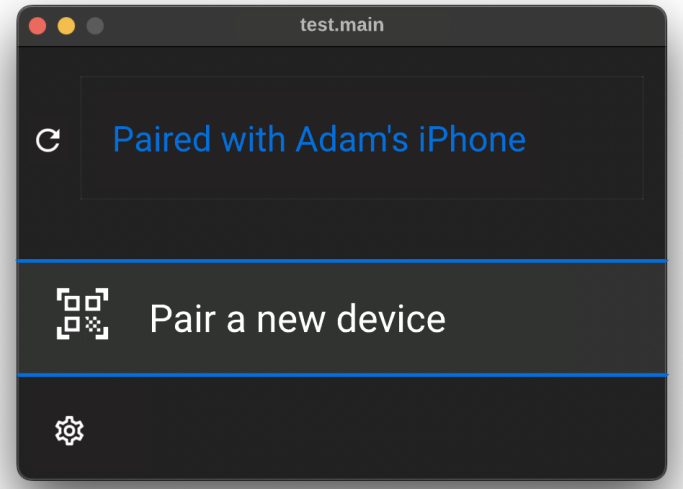
CorgKey w/ Connected Stub Website



Connected Stub Website Settings Dropdown



Desktop Client Home Screen



Desktop Client Pairing Screen

APPENDICES:

- Technologies Employed:
    - Flutter
        - Frontend implementation for mobile application, desktop client
        - State management & persistent storage of connected sites
    - Firebase Cloud Messaging
        - Transfer of information between mobile application, desktop client and web application
    - Python
        - Emulation of HID to act as virtual USB authenticator device within desktop client
    - WebAuthn API
        - Handle authentication challenges, relying party ID, attestations, assertions
    - Figma
        - Application planning & wireframing
    - Notion
        - Note-taking & organizational tools