EXHIBIT A

Unified Push Notification Project Scope and Requirements

The following milestones and related tasks shall be completed as part of this project:

1. **UnifiedPush Specification**
    - The full specification (Android + backend) shall be licensed with the Apache-2.0 license and hosted in a public git repository
    - Update the UnifiedPush backend specification to be compatible with RFC8030, RFC8291 and RFC8292
    - Make sure the onboard specification is fully compatible with the updated backend specification and supports all features from the RFCs (e.g., VAPID)
    - Create pull requests for all specification changes that allow for community feedback over a 2-week period
    - Improve onboard specification by specifying currently unanswered implementation details to clarify edge cases and to improve security including at least the following measures:
        - Specify behavior in case the distributor is currently offline
            - E.g., Behavior if the distributor is offline and must handle a unregister request and can't inform its push-server about the unregister event
        - Specify a rate limit mechanism to prevent apps from sending too many registration requests
            - E.g., Limit maximum number of allowed registrations for an app
            - E.g., Allow the distributor to throttle
        - Specify requirements for all onboard 'extras' like registration messages, message IDs etc. to allow for input validation
            - E.g., Maximum length of the registration message
            - E.g., Minimum requirements for message IDs and tokens
        - Specify Distributor behavior in case apps do not follow the requirements for the 'extras'
        - Clearly label MUST and SHOULD behavior for apps and distributors
        - Add a mechanism to allow Distributors to only accept requests from apps that support certain features
            - E.g., only accept VAPID authenticated requests
        - Clearly specify that apps MUST check the registration token to detect 'unauthenticated' requests
        - Clearly specify how message IDs can be handled securely to prevent apps from being able to acknowledge notifications on behave of other apps

2. **UnifiedPush Connector**
    - Provide an updated Android connector library that fully supports the new specification
    - The connector shall be licensed with the Apache-2.0 license and hosted in a public git repository
    - Create a pull request for all changes that allow for community feedback over a 2-week period
    - Provide an automated CI/CD to automatically build and test the UnifiedPush Connector
    - New versions of the connector shall be published to maven central
    - Add UnifiedPush support to the COVESA SDK which includes automatic decryption handling to simplify the user experience compared to the raw connector-library
        - For this a pull request on the COVESA SDK GitHub page shall be created to allow for community feedback on the integration over a 2-week period
    - Provide a simple Android project to show potential developers how to use UnifiedPush in an Android app following the best practices
        - Provide a version that details the usage of the UnifiedPush Connector (this could be the Example App)
        - Provide a version that shows the usage of the UnifiedPush COVESA SDK integration
    - Provide extensive documentation for the UnifiedPush connector as well as the COVESA SDK integration of the UnifiedPush Connector
    - Open a public GitHub discussion on Androids background work limitations on the COVESA SDK GitHub page
        - Discuss how background work limitations (esp. being prohibited from launching foreground services from the background) impacts apps

3. **UnifiedPush Example App**
    - Provide an updated example app that fully supports the new specification
    - The example-app shall be licensed with the Apache-2.0 license and hosted on a public git repository
    - Create a pull request for all changes that allow for community feedback over a 2-week period
    - Provide UI functionality to test all features supported by the new specification. Including:
        - Time-to-live
        - VAPID
        - Encryption
        - Topics
        - Urgency
        - OPTIONAL: Delivery Receipts

- Provide an automated CI/CD to automatically build and test new versions of the UnifiedPush Example app
- New versions of the example app shall be published to F-Droid
- Provide documentation for the example app and detail how users can use it to test distributors and push-servers

4. **Reference Implementation Push-Server**
    - Provide a reference implementation of a UnifiedPush push-server based on the new UnifiedPush specification
        - Based the implementation on Mozilla's autopush-rs
    - [OPTIONAL] The reference implementation supports the delivery receipts feature mentioned in RFC8030
    - Create pull requests for all necessary autopush-rs changes in the autopush-rs GitHub repository
    - Provide an automated CI/CD that tests and builds up-to-date artifacts of the push-server to allow self-hosting
    - Provide a simple way (one-command) to self-host the modified Mozilla push-server on a desktop Linux machine
        - E.g., with a docker-compose file and publicly hosted reference images
    - Provide extensive documentation on how to run the potentially modified autopush-rs push-server on a desktop Linux and how to connect it with the new distributor

5. **Reference Implementation Distributor**
    - Provide an Android distributor fully compatible with the updated UnifiedPush specification
    - [OPTIONAL] The reference implementation supports the delivery receipts feature mentioned in RFC8030
    - The custom distributor shall be licensed with the Apache-2.0 license and hosted in a public git repository
    - Create a pull request for all changes that allow for community feedback over a 2-week period
    - Allow the distributor to connect to the public Mozilla autopush-rs servers (if Mozilla does not block unknown clients from connecting)
    - Provide a CI/CD to automatically build and test new versions of the distributor
    - New versions of the distributor shall be published to F-Droid
    - Provide extensive documentation on the distributor
        - Detail design considerations
        - Detail the core code that is needed for a specification compliant distributor (only the service, without any UI)
    - Provide a mechanism to change the minimum urgency required for notifications to wake up apps based on a signal

        - E.g., The signal could be the battery power and connectivity state of the device
    - Provide a UI for the distributor which supports at least the following features:
        - Change the used push-server in the distributor UI
        - See current subscriptions (application name and package name)
        - Unsubscribe individual subscriptions

Unified Push Notification Project Time Plan

1. UnifiedPush Specification:
    a. Implementation: September 2nd, 2024 – September 20th, 2024
    b. Review Period: September 20th – October 4th
2. UnifiedPush Connector & UnifiedPush Example App:
    a. Implementation: September 23rd, 2024 – October 25th, 2024
    b. Review Period: November 8th, 2024 – November 22nd, 2024
3. Reference Implementation:
    a. Implementation: October 28th, 2024 – December 20th, 2024
    b. Review Period: December 23rd, 2024 – January 17th, 2025