**Email Helper App Documentation**

**Overview:**

Most of the overview and information about the goals for the app can be found in the Email Helper app design pdf.

**Outside Features:**

This app was implemented using Javamail, which offers many features for searching, sending, and receiving emails using the Java programming language. It also uses SQLite databases to store the emails in the app private data.

**Main Features:**

* The app is able to make connections to IMAP/SMTP mail accounts to retrieve emails, both when the app is opened, with a “pull to refresh” feature on the ConversationFragment, and an auto updater.
* The app can send emails using SMTP login info.
* The app stores the connection data in the Shared Preferences of the app, so that they are persistent.
* The user is able to change their email address to a different mail account using those in the code or with if they specify the IMAP/SMTP servers
* The user is able to “subscribe” to email addresses and only receive emails from those subscribed accounts.
* The user is able to see their email conversations similar to an instant messaging app, where messages are organized by email address in their own fragments.
* The user is able to send an email using the SMS style messaging at the bottom of the conversation list, and these messages will show up on the screen and save to the database.
* Notifications that display when the app is not in use or display as a Toast when the app is in use. The notifications are configured inside the app and can be contact individualized.
* The User can display a personal QR code or scan others to add them to the subscription list
* Contacts can be gotten directly from the inbox using the “Show All Contacts from Inbox” checkbox. This list is sorted by number of emails received from that address in the last three months.
* Main screen with conversations updates to include the latest date of the contact was updated by creation or a new email.

**Future TODOs:**

* The app cannot currently show all attachments.
* There is no PGP or MIME decryption implemented.
* OAuth2.0 - If we implemented it it would be usable to the standard google account. When enabling the app to use OAuth we will need a client ID and possibly a client secret which is gotten from Google when you register the app with them. Contents will need to be added to the AuthenticationClass. I added some commented out variables into AuthenticationClass for this in the future. Hopefully, this will be able to be wrapped into a class/module.

**Issues/Bugs:**

No known bugs

-There is an issue with some attachments(non-image) being open.

-We are a less secure app in the eyes of google which is an issue.

**Ideas/Suggestions/Recommendations ( BY DSHADE Summer 2018)**

**Almost There**

**-Sounds with Notifications**

There is a bug in the android code for this version 28 and above but otherwise it works.

**-Attachments**

Currently, photo attachments are taken care of. Other attachments are just downloaded. There is work to send them off via intent but they get blocked.

**Not there yet**

**-Multiple emails**

Allow for multiple emails to be added much like gmail so we pull from multiple inboxes.

**-Encryption**

Encryption is not currently handled in the app.

**Non-Programing Recommendations**

-Rebrand EmailHelper to EmailToText

* I think this will give better visibility and slogans and emphasis on what the app does.

|  |  |  |
| --- | --- | --- |
| API | Cumlative distribution | Note |
| 10 | 0.3% |  |
| 15 | 0.7% |  |
| 16 | 2.2% | Current Lowest excluding 0.7% of devices |
| 17 | 4.4% |  |
| 18 | 5.0% |  |
| 19 | 15.3% | New Lowest excluding only 5% of devices. |
| 21 | 20.1% |  |
| 22 | 37.7% |  |

-Update lowest API from 16 to 19. View chart below for details