MSWE 264

Telegram

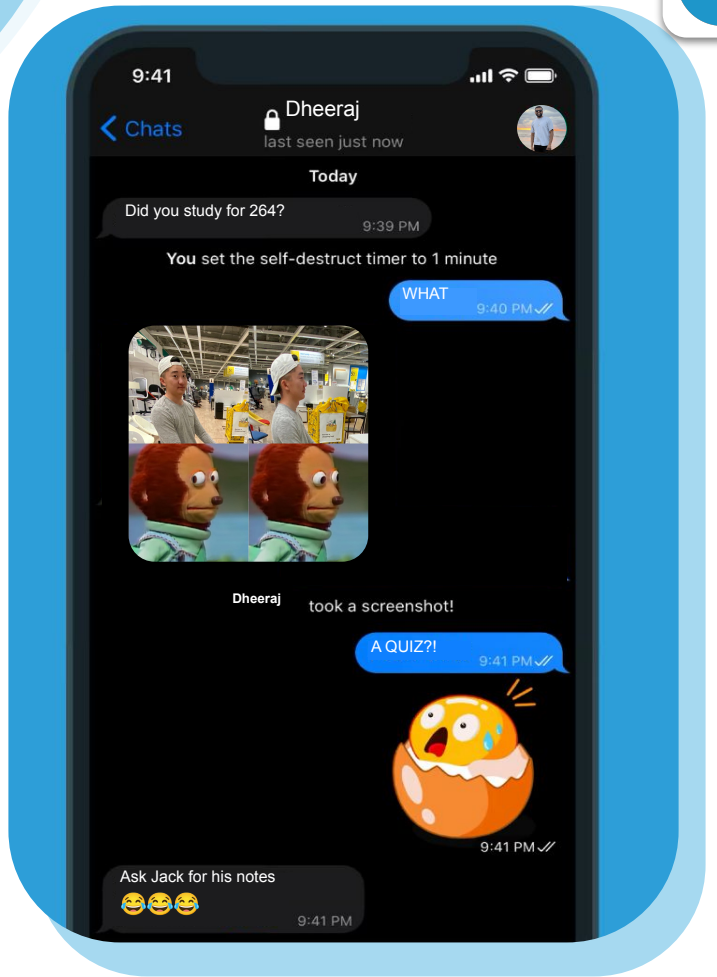
Analyzing Architecture and Design

What is Telegram?

Cross-platform, cloud based, messaging application with over 700 million monthly users

Create channels and chat groups of up to 200,000 people

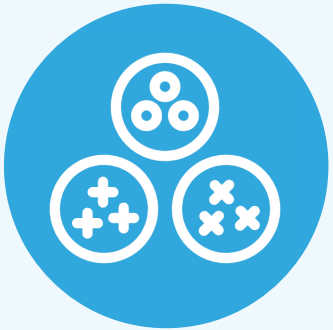
Make video and audio calls either 1:1 or conference style



Methodology

1

List out all activities in a table and write brief description of each activity



2

Group and map activities that are related to one another

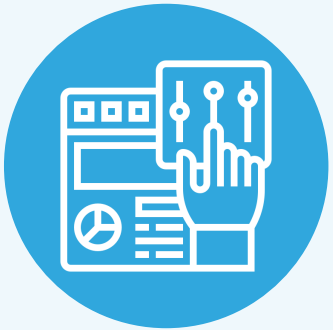
3

Filter out grouped activities that are not relevant to the main chat features

Methodology

4

Use Manifest.XML to list Services, Broadcast receivers, and Content Providers



5

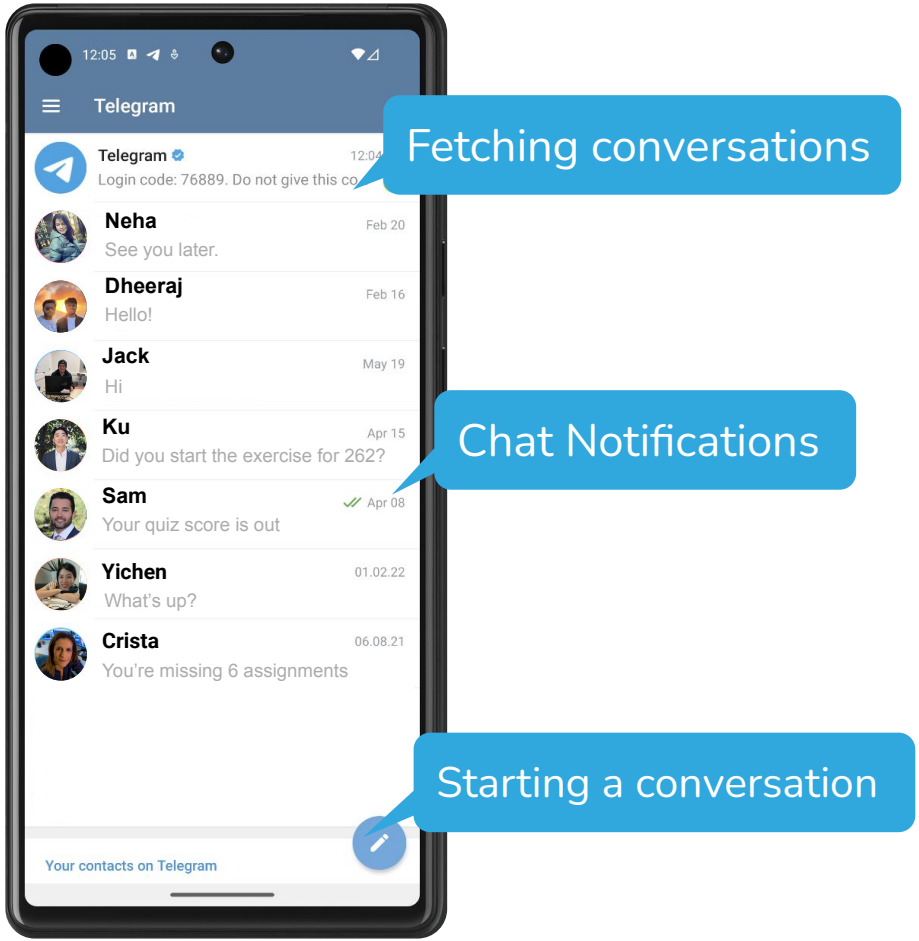
Find how these grouped activities interact with Services, Content Providers, and Broadcast Receivers

6

Finalize connectors between the activities, services, content providers, and broadcast receivers

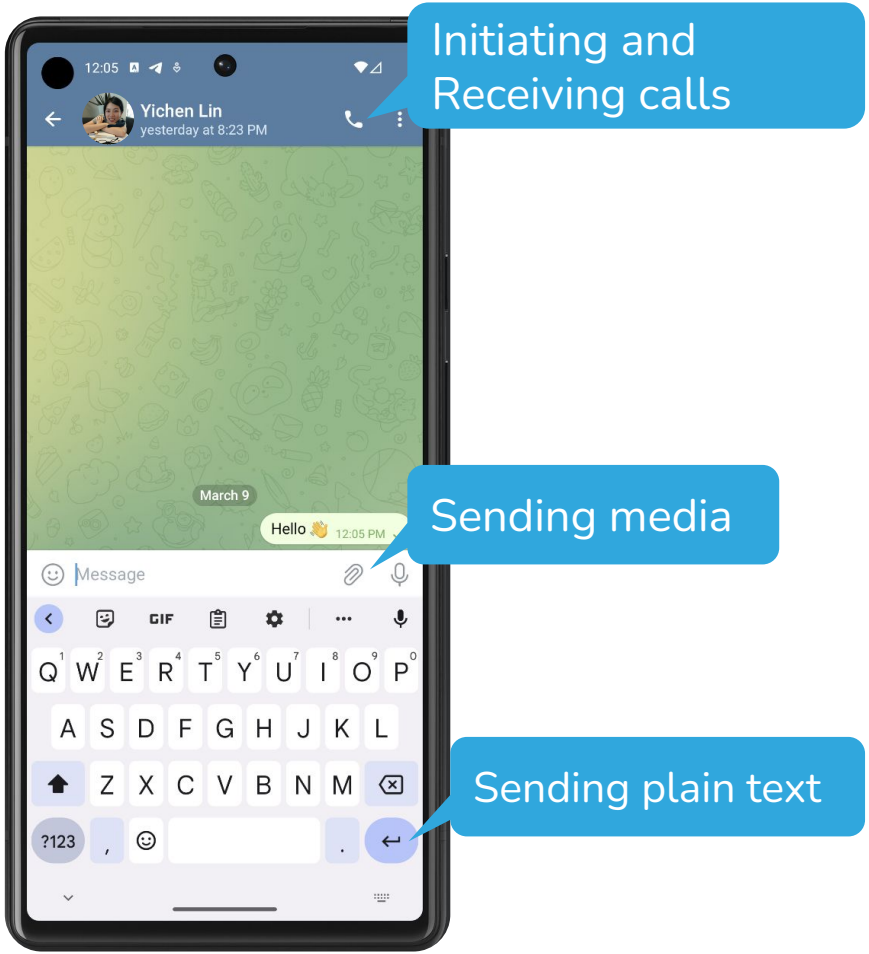
Yes #NoFilter

- Loading all existing chats and messages for the user to see upon launching the application
- Retrieving all messages inside of Conversation
- Starting a conversation when the pencil button is pressed



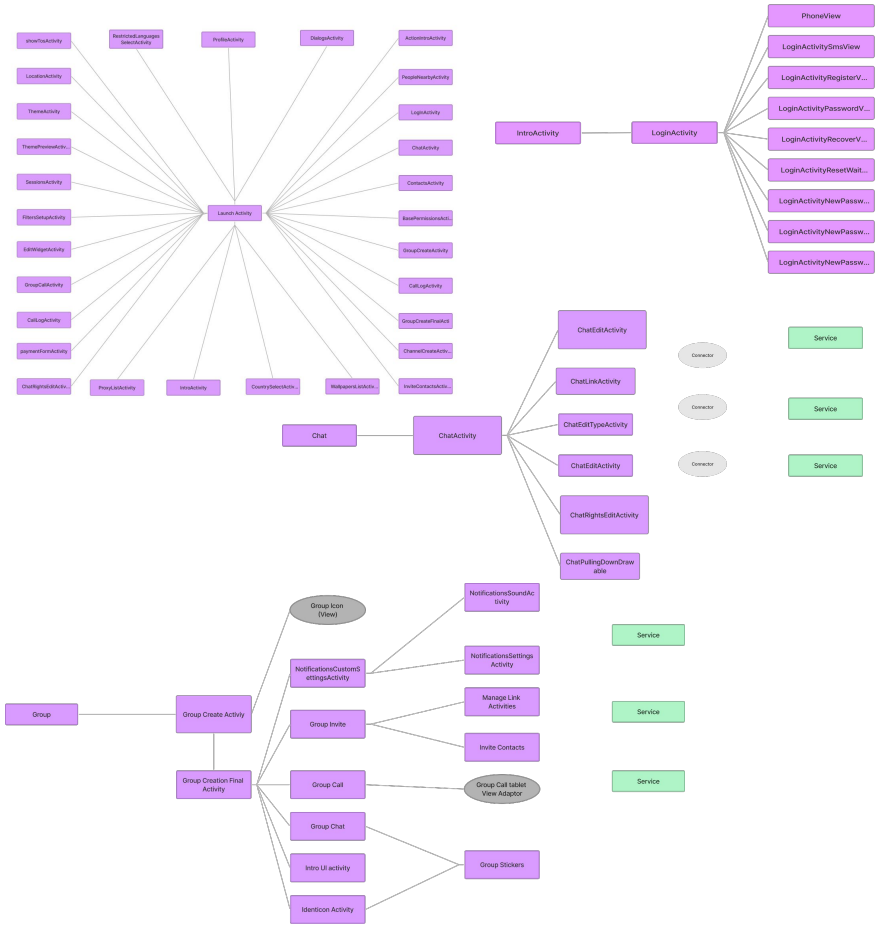
Yes ~~#NoFilter~~

- Initiating and receiving a voice call upon clicking the phone button
- Attaching media to a message by pressing on attachment
- Sending plain text



Solving the Maze

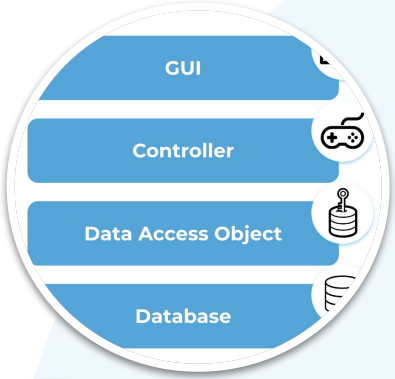
- Start at the chat activity and find method calls and any related activities
- Look at the Manifest to list out all services, Broadcast receivers, content providers, and intent filters
- Layout the initial configuration, and fill in connectors

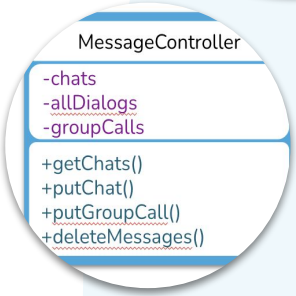


Architectural Styles

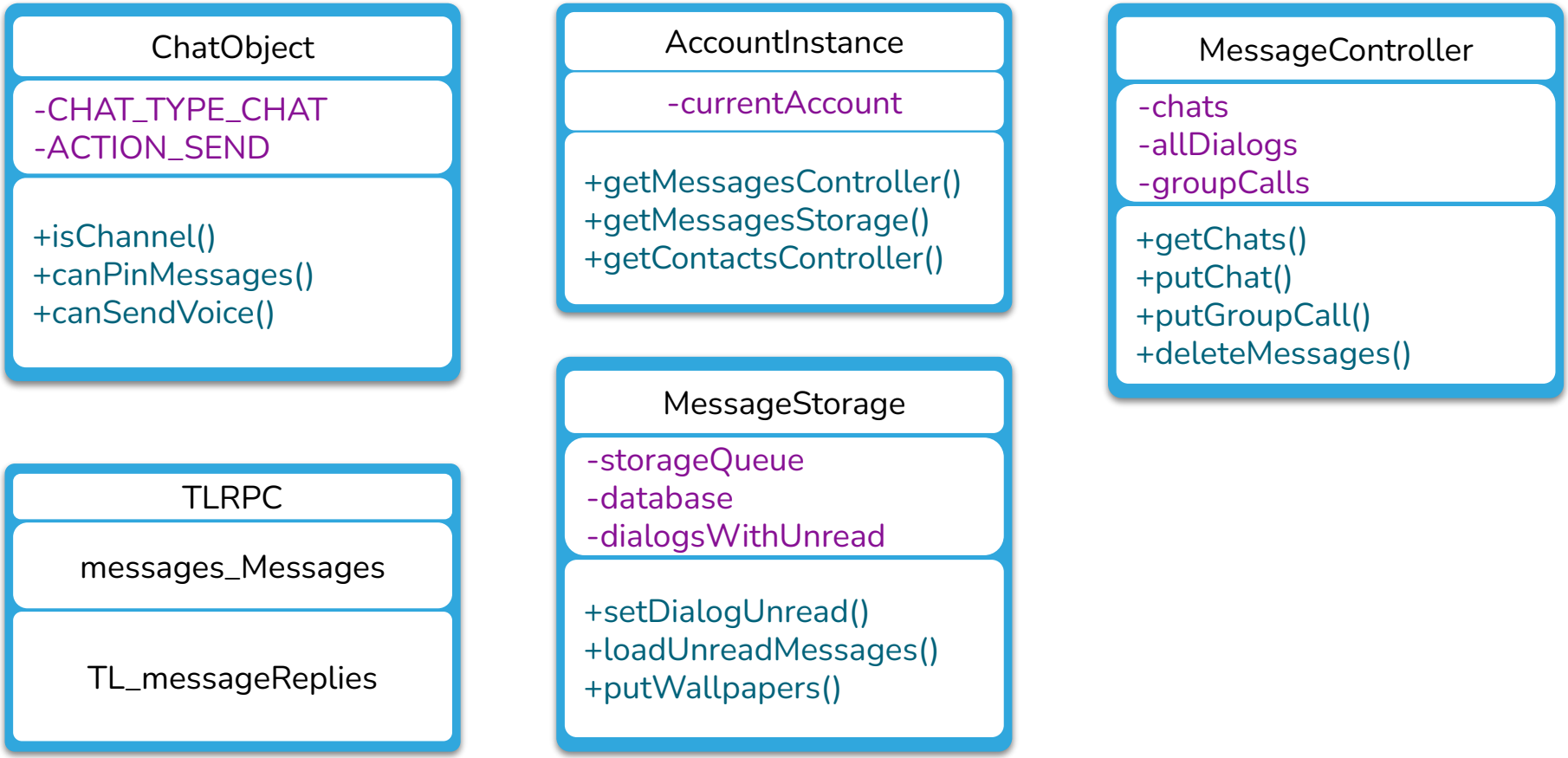
Architectural styles in Telegram:

- Object-Oriented Style
- Layered Style
- Client-Server Style
- Publish-Subscribe Style
- Message-Based Explicit Invocation
- Message-Based Implicit Invocation

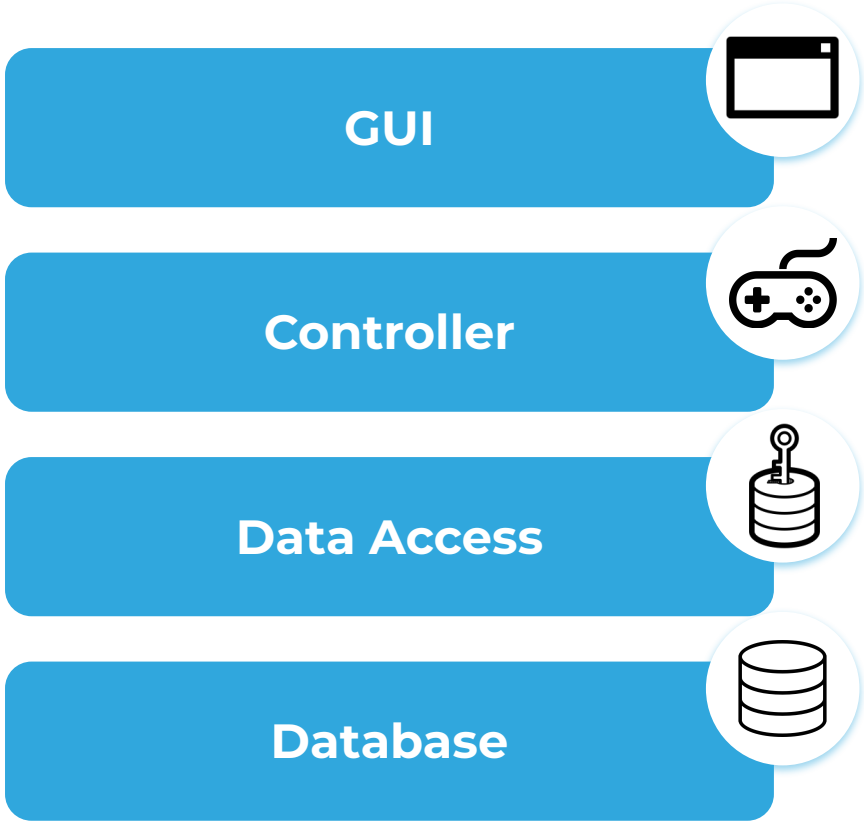




Object-Oriented Style



Layered Style



Layered Style

ThemePreviewActivity.java (GUI)

```
MessagesController.getInstance(currentAccount).saveWallpaperToServer(path, wallpaperInfo, slug != null, 0);
```

MessageController.java (Controller)

```
public void saveWallpaperToServer(File path, Theme.WallpaperInfo info) {
getMessagesStorage().putWallpapers(arrayList, -3);
}
```

MessageStorage.java (Data Access)

```
public void putWallpapers(ArrayList<TLRPC.WallPaper> wallPapers, int action) {
state=database.executeFast("UPDATE wallpapers2 SET data = ? WHERE uid = ?");
```

SQLite (Database)

Layered Style

ThemePreviewActivity.java (GUI)

MessageController.java (Controller)

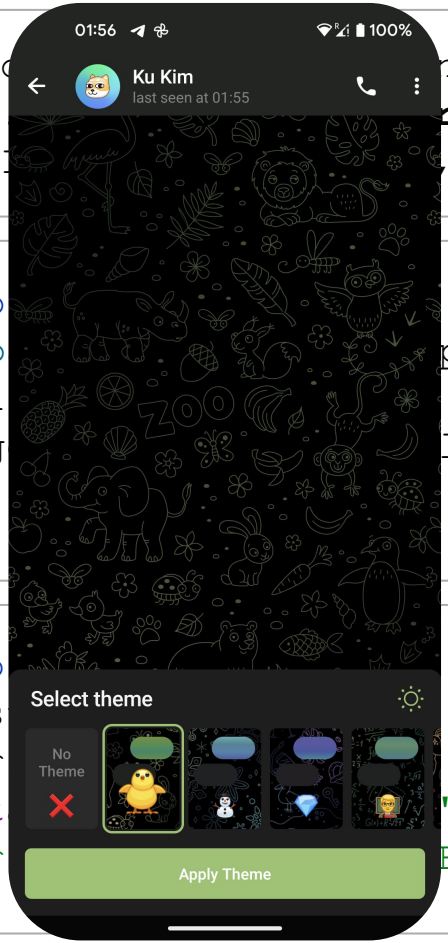
MessageStorage.java (Data Access)

SQLite (Database)

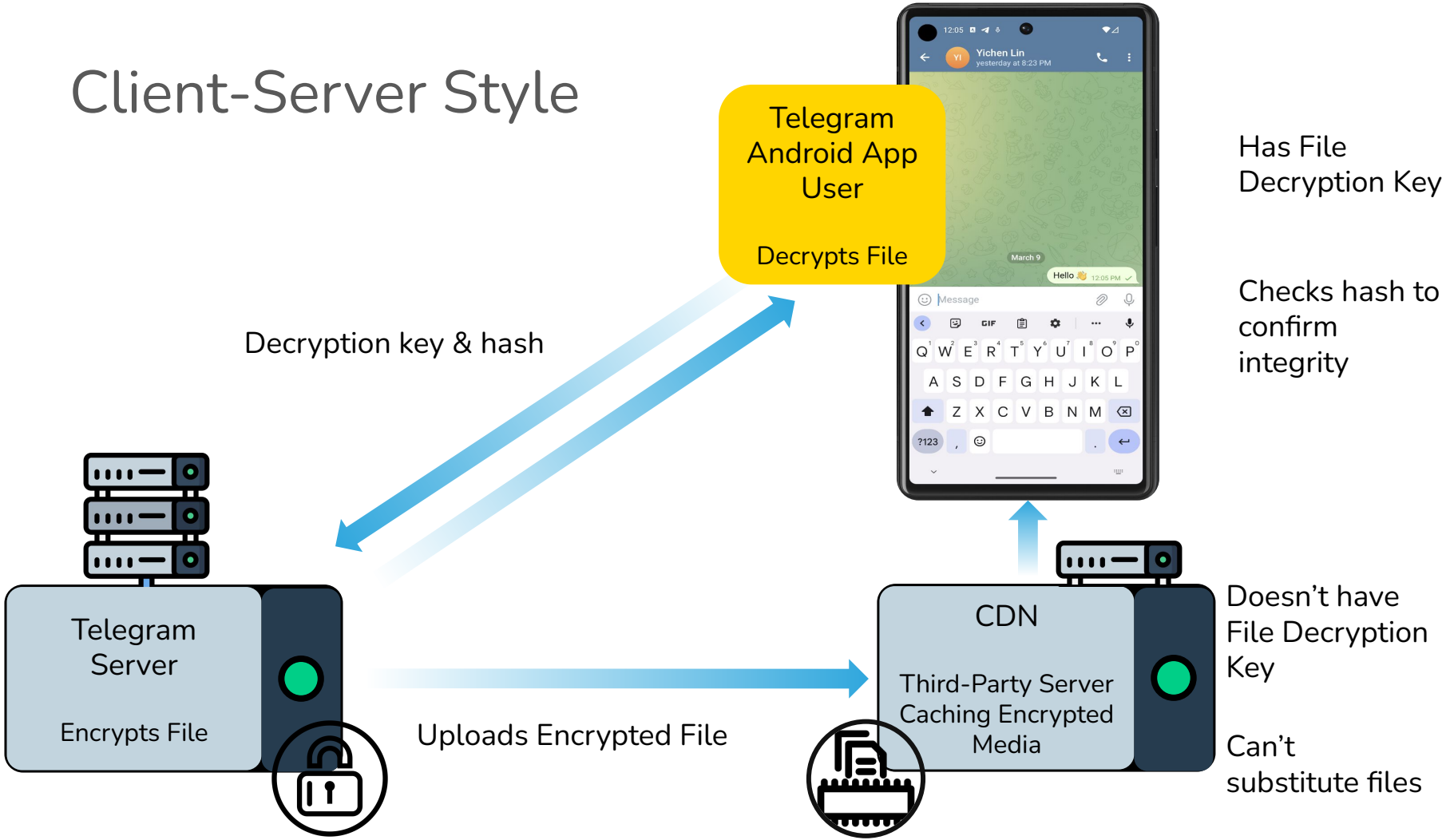
```
MessagesC                                    nce(curren
tAccount)                                    rver(path,
wallpaperI                                     0);

public vo
saveWallp                                    path,
Theme.Wal
getMessag                                    lpapers(ar
rayList,
}

public vo
(ArrayLis
wallPaper
state=dat                                    "UPDATE
wallpaper                                    ERE uid =
?");
```



Client-Server Style



Publish-Subscribe Style

```
if (threadMessageId == 0) {
    getNotificationCenter().addObserver(this, NotificationCenter.screenshotTook);
    getNotificationCenter().addObserver(this, NotificationCenter.encryptedChatUpdated);
    getNotificationCenter().addObserver(this, NotificationCenter.messagesReadEncrypted);
    getNotificationCenter().addObserver(this, NotificationCenter.botKeyboardDidLoad);
    getNotificationCenter().addObserver(this, NotificationCenter.updateMentionsCount);
    getNotificationCenter().addObserver(this, NotificationCenter.newDraftReceived);
    getNotificationCenter().addObserver(this, NotificationCenter.chatOnlineCountDidLoad);
    getNotificationCenter().addObserver(this, NotificationCenter.peerSettingsDidLoad);
    getNotificationCenter().addObserver(this, NotificationCenter.didLoadPinnedMessages);
    getNotificationCenter().addObserver(this, NotificationCenter.commentsRead);
    getNotificationCenter().addObserver(this, NotificationCenter.changeRepliesCounter);
    getNotificationCenter().addObserver(this, NotificationCenter.messagesRead);
    getNotificationCenter().addObserver(this, NotificationCenter.didLoadChatInviter);
    getNotificationCenter().addObserver(this, NotificationCenter.groupCallUpdated);
} else {
    getNotificationCenter().addObserver(this, NotificationCenter.threadMessagesRead);
    if (isTopic) {
        getNotificationCenter().addObserver(this,
NotificationCenter.updateMentionsCount);
        getNotificationCenter().addObserver(this, NotificationCenter.didLoadPinnedMessages);
    }
```

Publish-Subscribe Style

Publisher

NotificationCenter

Events

NotificationCenter.screenshotTook

NotificationCenter.encryptedChatUpdated

NotificationCenter.messagesRead

Subscriber

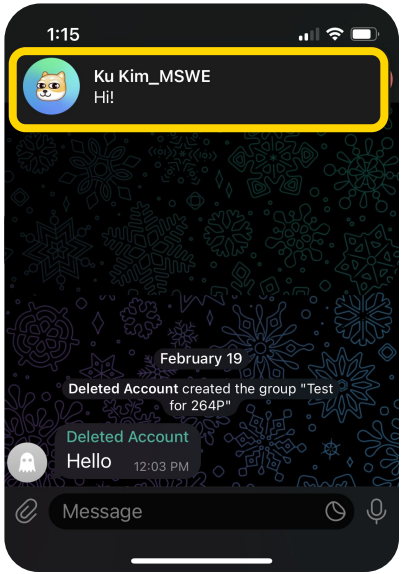
ChatActivity

Message-based Explicit Invocation Style

ChatActivity

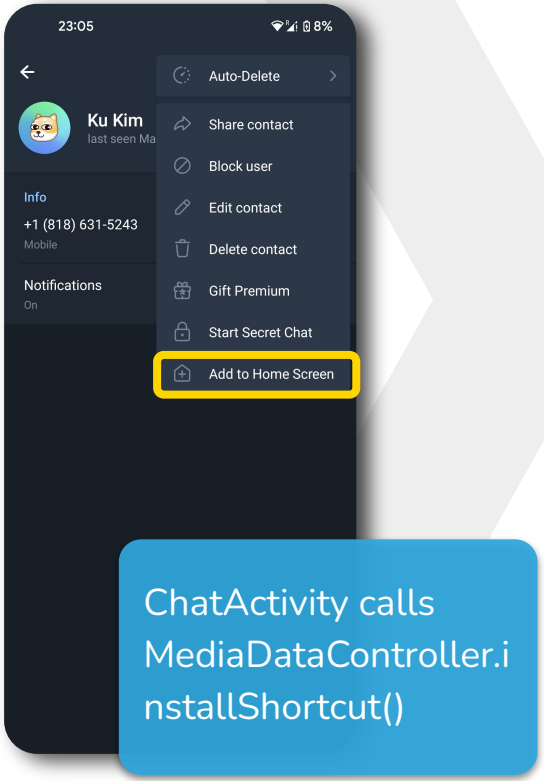
NotificationController

PopupNotification Activity

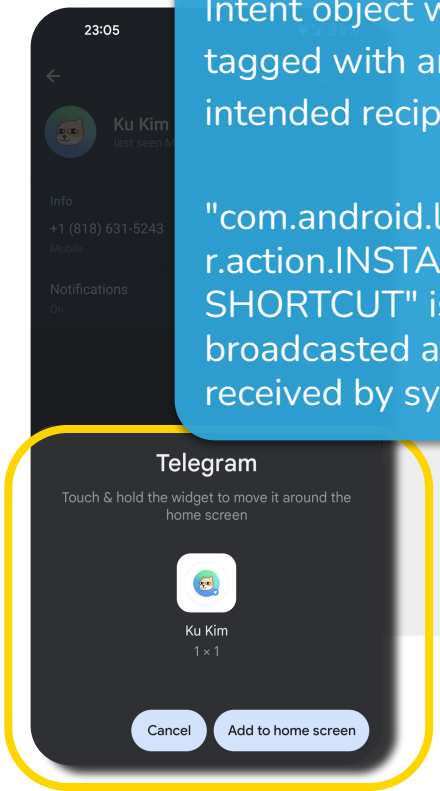



```
Intent popupIntent = new Intent(ApplicationLoader.applicationContext,
PopupNotificationActivity.class);
popupIntent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK |
Intent.FLAG_ACTIVITY_NO_ANIMATION | Intent.FLAG_ACTIVITY_NO_USER_ACTION |
Intent.FLAG_FROM_BACKGROUND);
try {
    ApplicationLoader.applicationContext.startActivity(popupIntent);
} catch (Throwable ignore) {}
```

Message-based Implicit Invocation Style

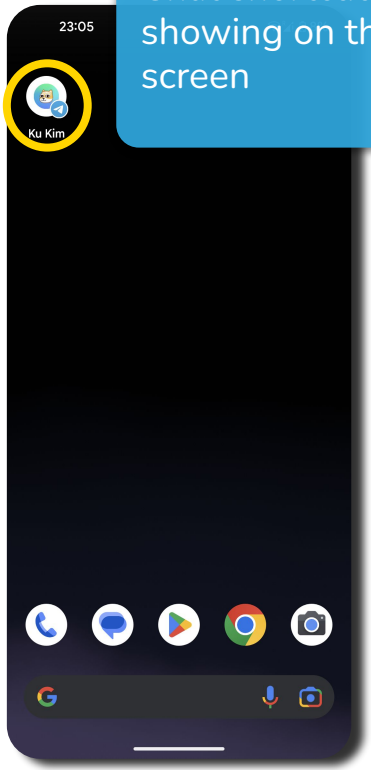


ChatActivity calls MediaDataController.installShortcut()



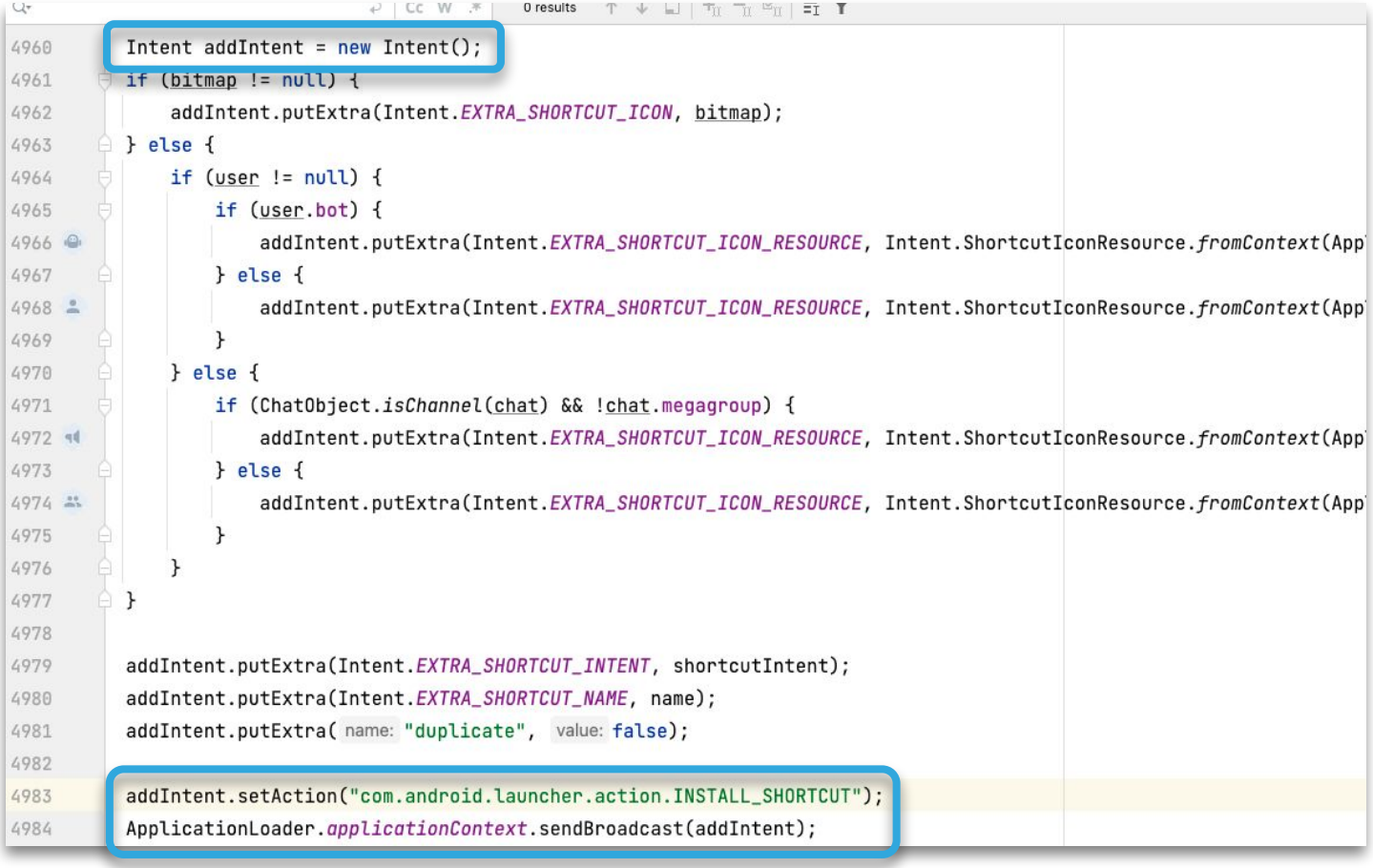
Intent object without tagged with any intended recipients

"com.android.launcher.action.INSTALL_SHORTCUT" is broadcasted and received by system



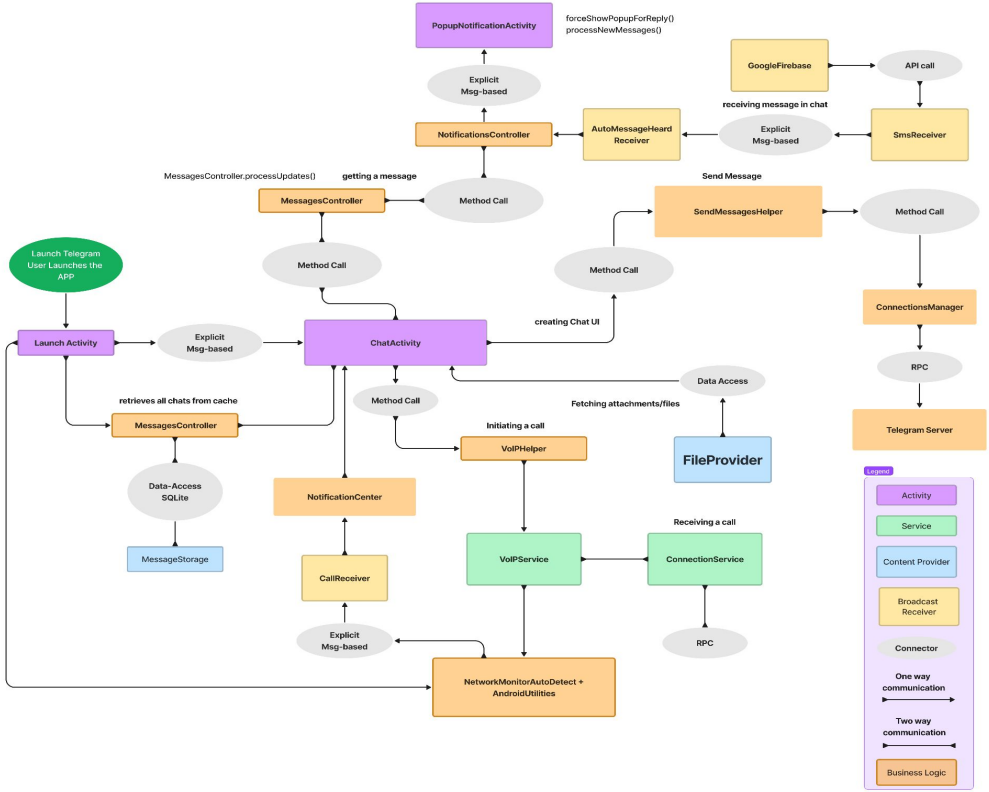
Chat shortcut showing on the screen

Message-based Implicit Invocation Styles



```
4960  Intent addIntent = new Intent();
4961  if (bitmap != null) {
4962      addIntent.putExtra(Intent.EXTRA_SHORTCUT_ICON, bitmap);
4963  } else {
4964      if (user != null) {
4965          if (user.bot) {
4966              addIntent.putExtra(Intent.EXTRA_SHORTCUT_ICON_RESOURCE, Intent.ShortcutIconResource.fromContext(App
4967          } else {
4968              addIntent.putExtra(Intent.EXTRA_SHORTCUT_ICON_RESOURCE, Intent.ShortcutIconResource.fromContext(App
4969          }
4970      } else {
4971          if (ChatObject.isChannel(chat) && !chat.megagroup) {
4972              addIntent.putExtra(Intent.EXTRA_SHORTCUT_ICON_RESOURCE, Intent.ShortcutIconResource.fromContext(App
4973          } else {
4974              addIntent.putExtra(Intent.EXTRA_SHORTCUT_ICON_RESOURCE, Intent.ShortcutIconResource.fromContext(App
4975          }
4976      }
4977  }
4978
4979  addIntent.putExtra(Intent.EXTRA_SHORTCUT_INTENT, shortcutIntent);
4980  addIntent.putExtra(Intent.EXTRA_SHORTCUT_NAME, name);
4981  addIntent.putExtra( name: "duplicate",  value: false);
4982
4983  addIntent.setAction("com.android.launcher.action.INSTALL_SHORTCUT");
4984  ApplicationLoader.applicationContext.sendBroadcast(addIntent);
```

Architecture of Telegram Chat



Architecture graph breakdown

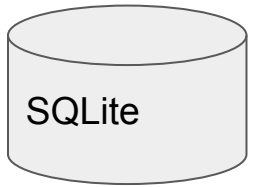
Functionalities:

- Sending a message
- Receiving a message
- Retrieving a chat
- Initiating a call
- Receiving a call

Android architecture components explored:

- Content Providers
- Broadcast Receivers
- Connectors
- Activity
- Services

Shared State Style



Publish and Subscribe Style

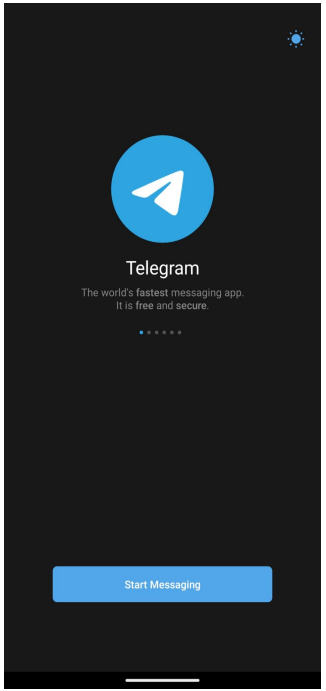
Activity observers

Observer

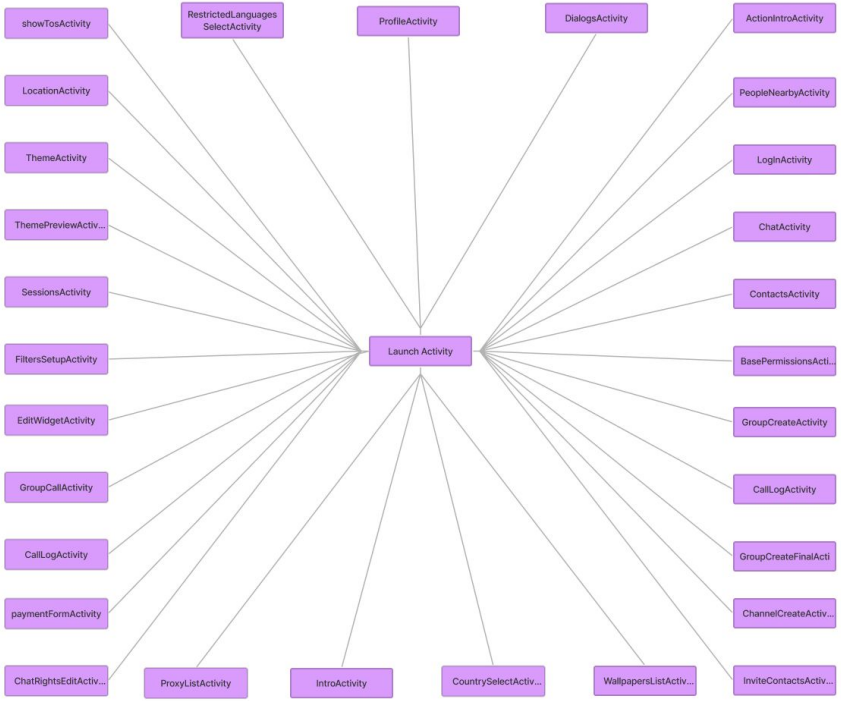
Notification

User Interface - Launch Activity

- Contact fetching
- UI Loading
- Theme Fetching
- Loading previous messages, group chats, call logs



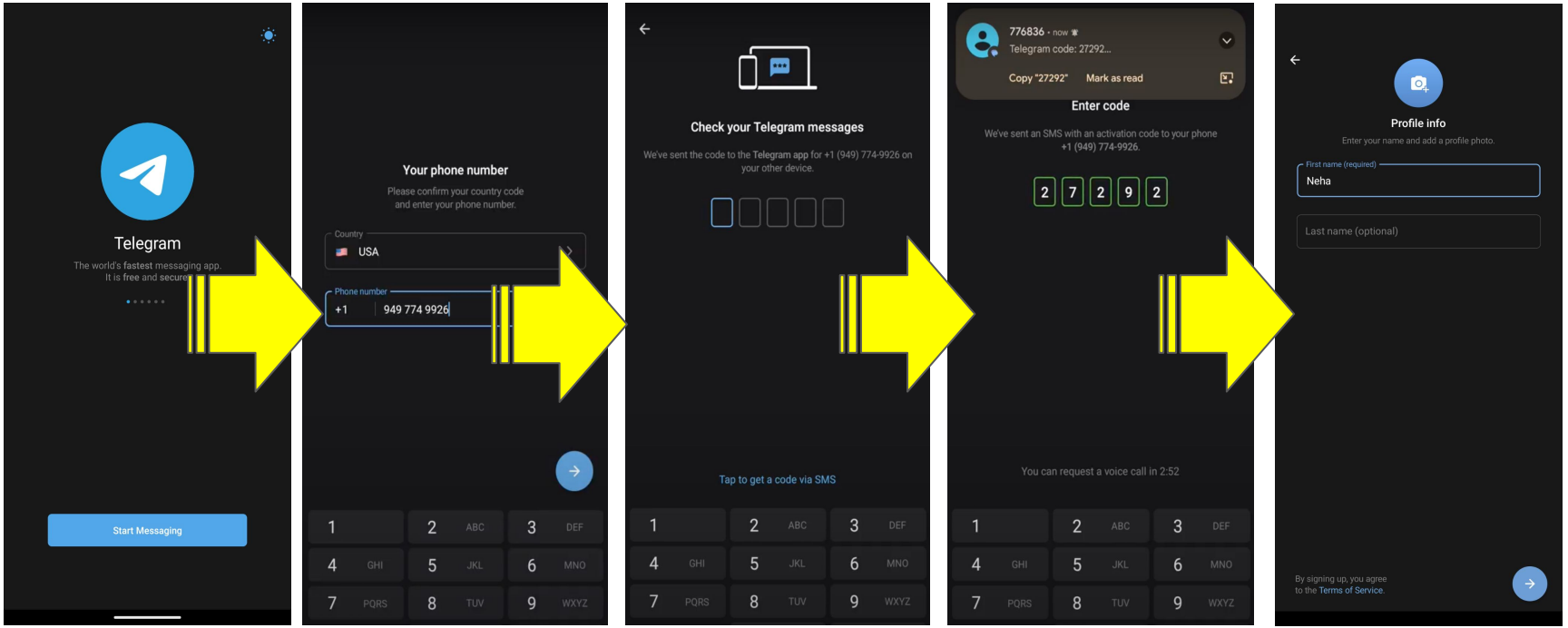
Launch activity graph



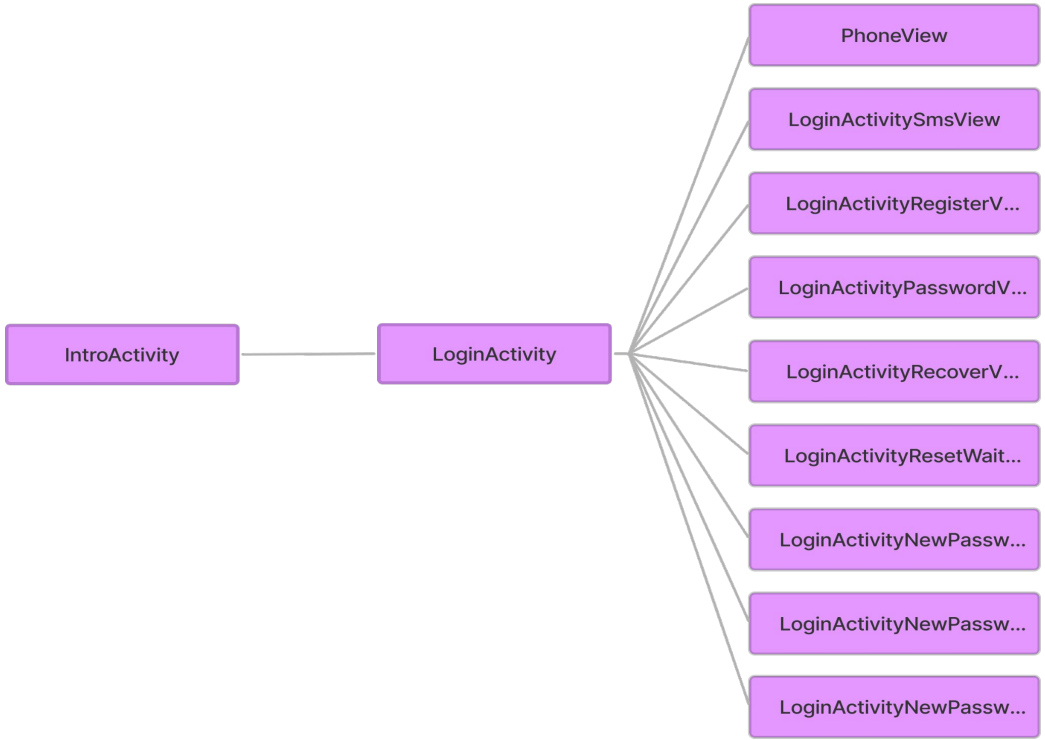
User Interface - User Account Setup

1. Intro Activity
2. Login Activity
    a. PhoneView
    b. LoginActivitySmsView
    c. LoginActivityRegisterView
    d. LoginActivityPasswordView
    e. LoginActivityRecoverView
    f. LoginActivityResetWaitView
    g. LoginActivityNewPasswordView
    h. LoginActivitySetupEmail
    i. LoginActivityEmailCodeView

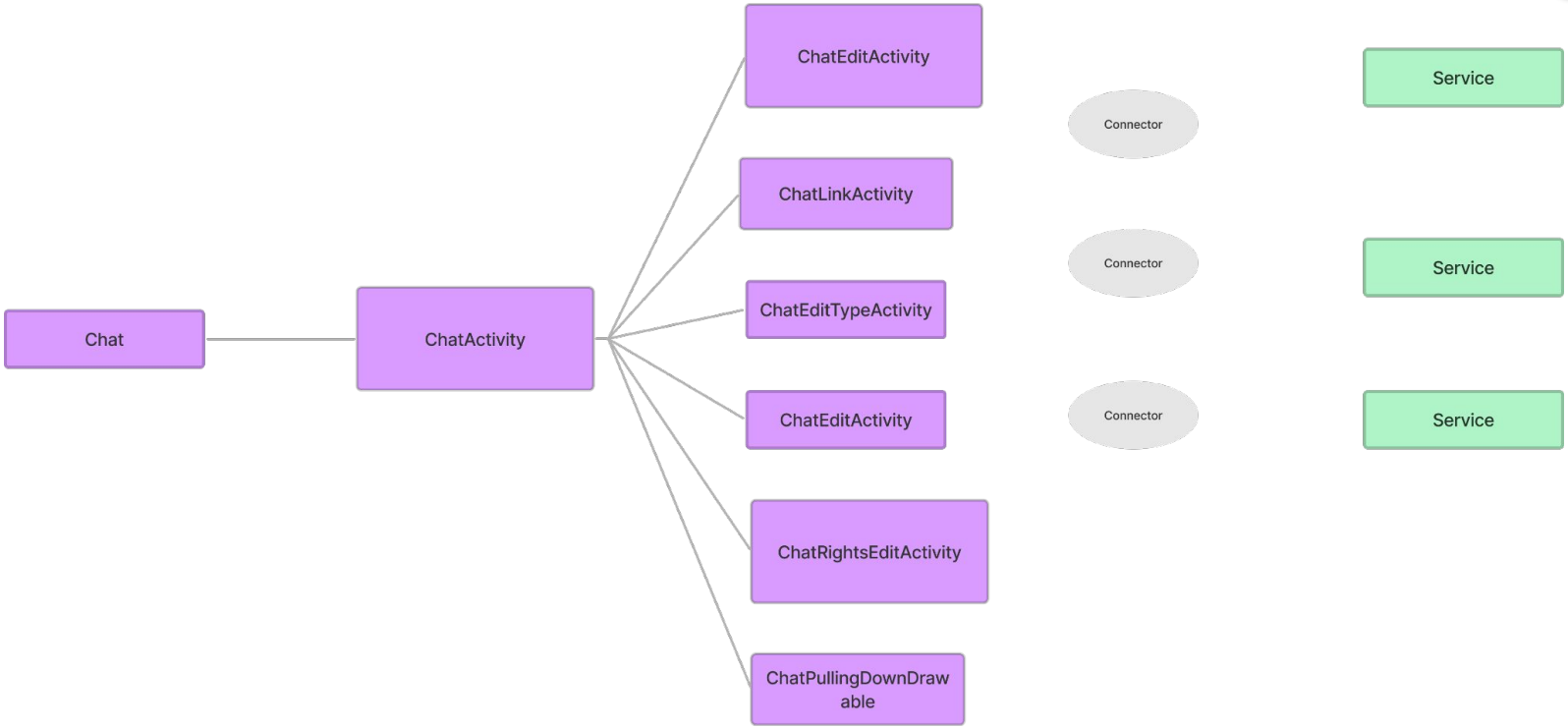
User Interface - User Account Setup



User Account Setup Graph

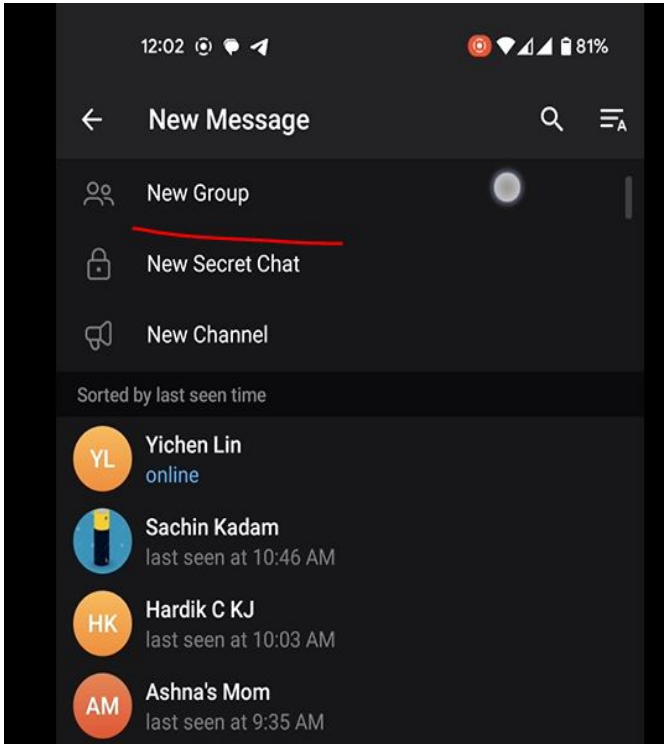


Chat Graph


Chat
ChatActivity
ChatEditActivity
ChatLinkActivity
ChatEditTypeActivity
ChatEditActivity
ChatRightsEditActivity
ChatPullingDownDrawable
Connector
Connector
Connector
Service
Service
Service

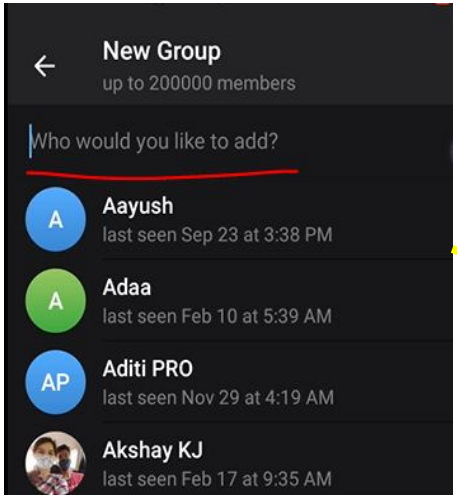
User Interface - Creating Groups

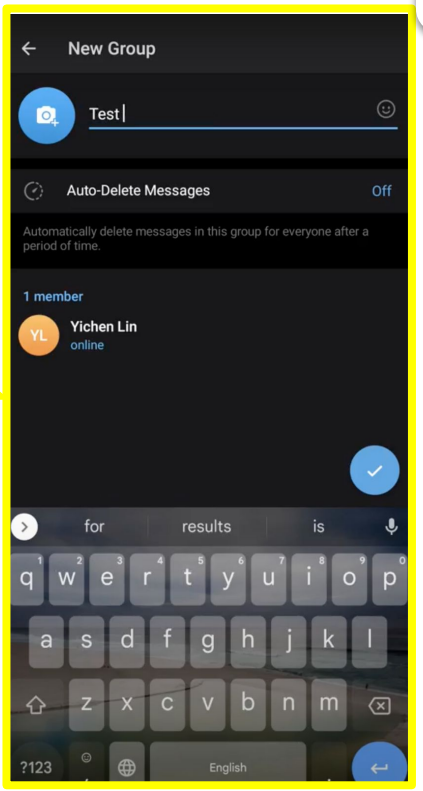
- Fetching list of Contacts
- Fetching messages in a conversation
- User selects new group option



User Interface - Creating Groups

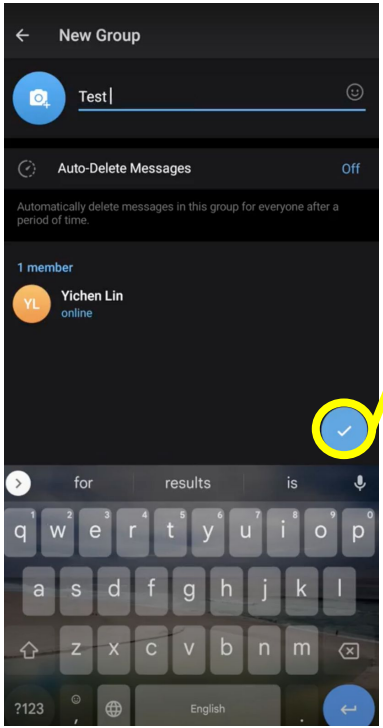
- Add Contacts into the group
- Contact list gets loaded
- User can select the contacts that need to be added

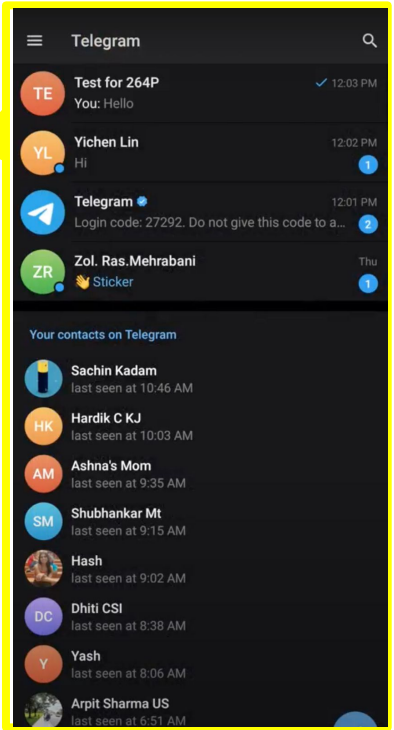




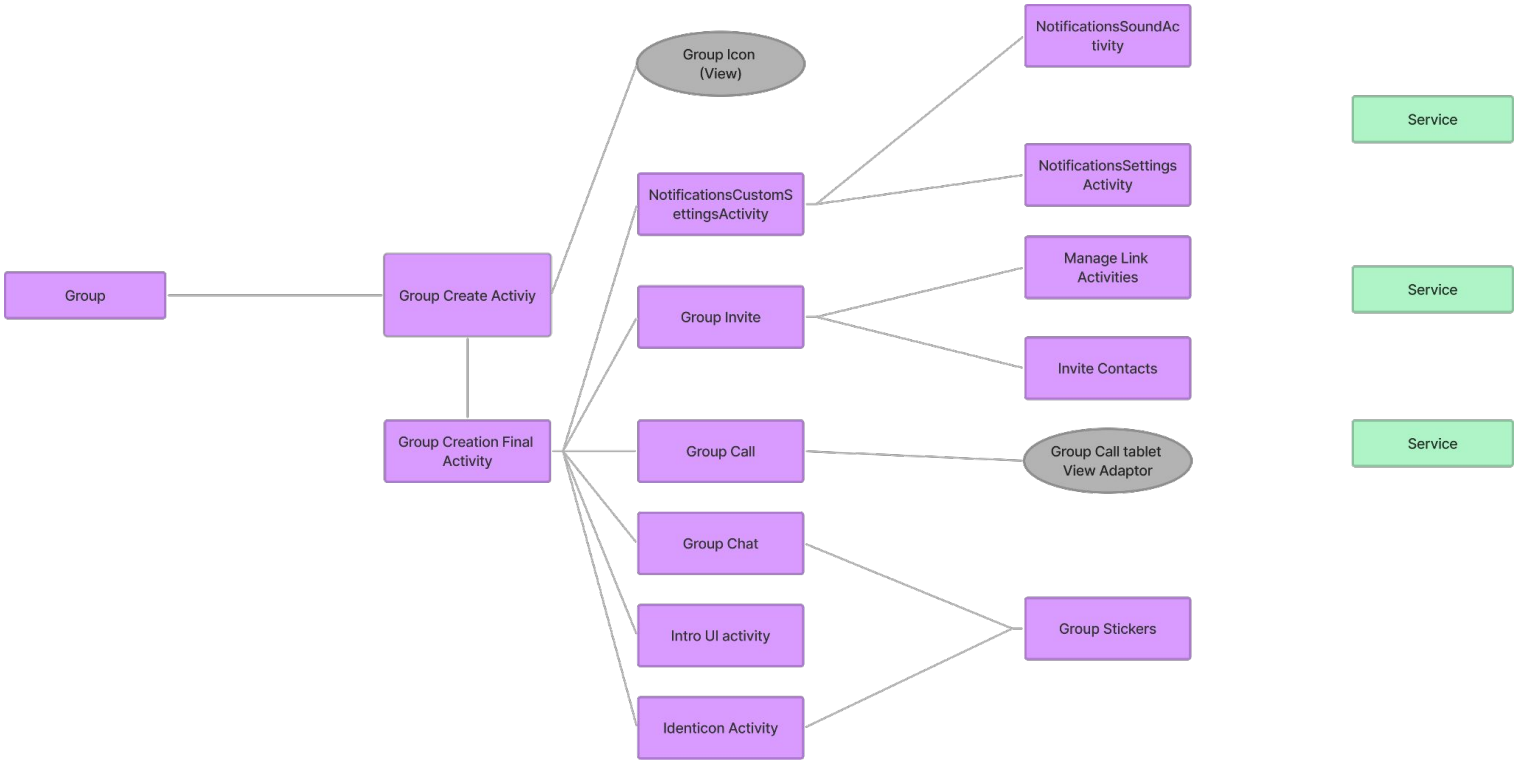
User Interface - Creating Groups

- User provides group Name
- Group gets created
- User also can add participants to group via an invite link
- Group provides almost all functionalities as the direct message (DM) room provides





Components Involved in Group Creation Component/feature



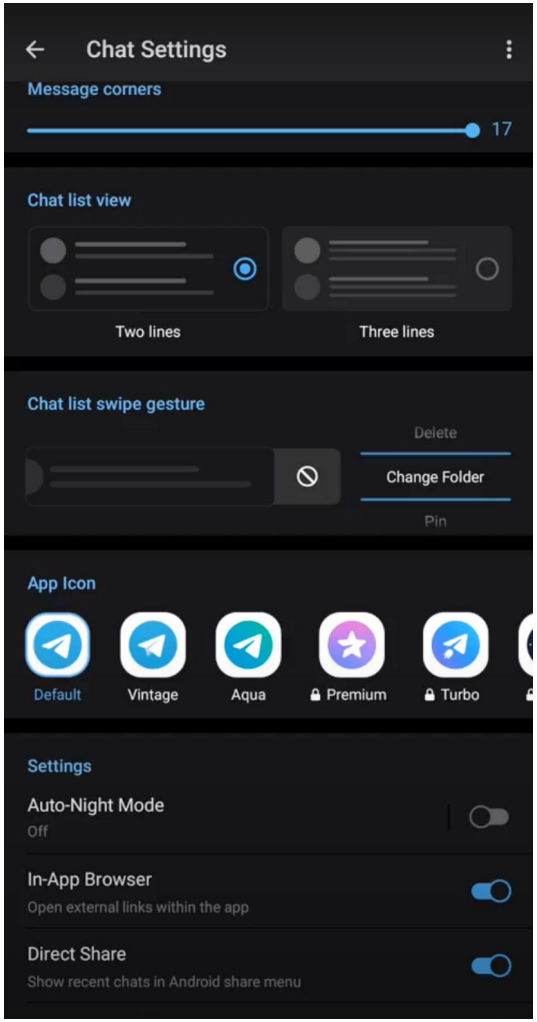
Conclusion and Future Direction

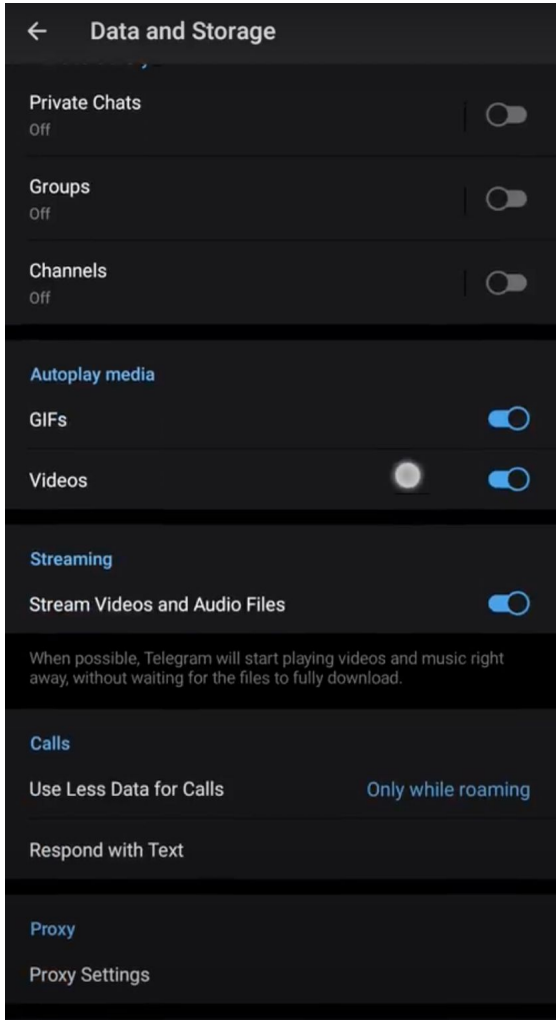
- Launch Activity
- User Account Setup
- Chat
- Group activities

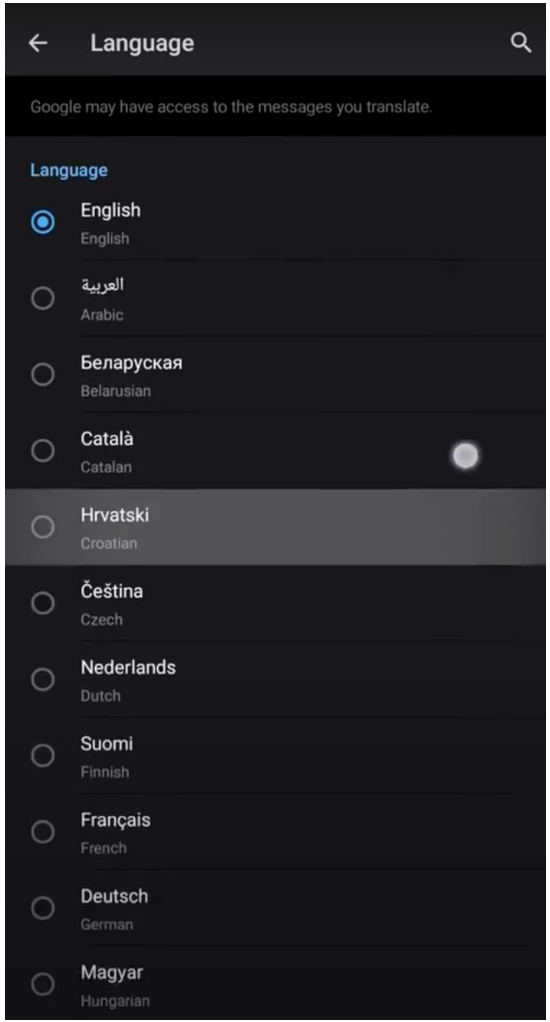
Retrieving the interactions between the UI and Database in more thorough detail.

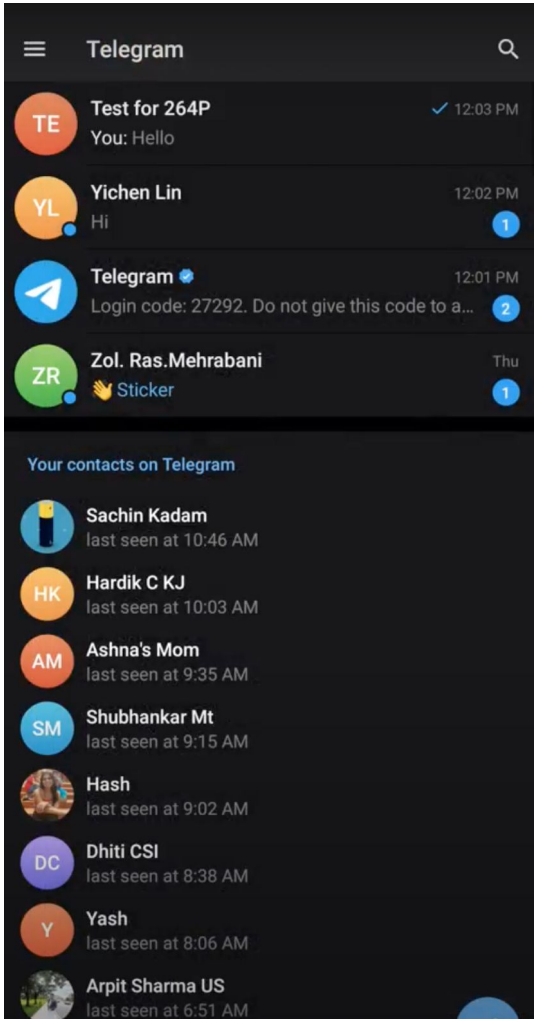
Finding connectors between these larger components and finding the points of interaction.

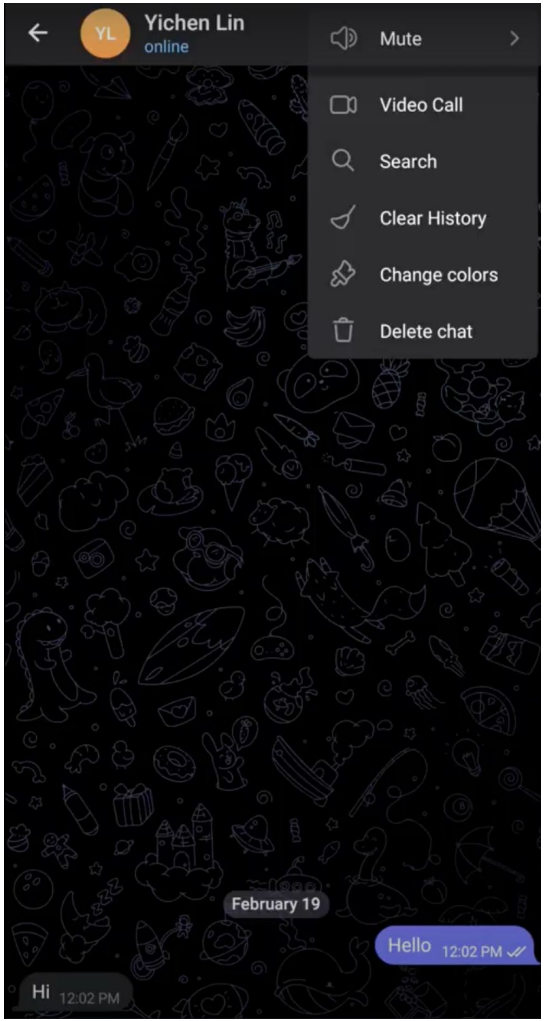
Coming up with a representation of architecture for these activities and also Telegram app as a whole .

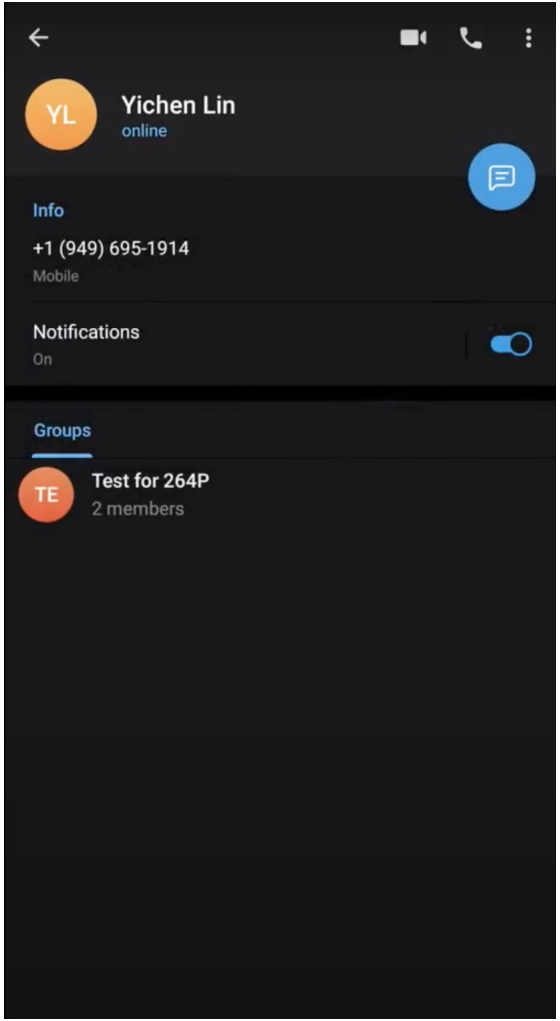

Chat Settings
Message corners
17
Chat list view
Two lines
Three lines
Chat list swipe gesture
Delete
Change Folder
Pin
App Icon
Default
Vintage
Aqua
Premium
Turbo
Settings
Auto-Night Mode
Off
In-App Browser
Open external links within the app
Direct Share
Show recent chats in Android share menu

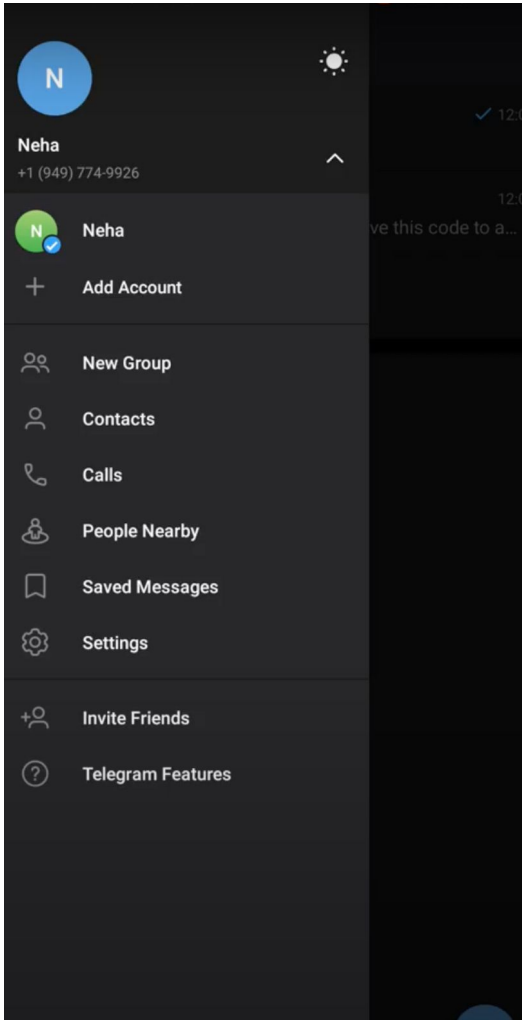

Data and Storage
Private Chats
Off
Groups
Off
Channels
Off
Autoplay media
GIFs
Videos
Streaming
Stream Videos and Audio Files
When possible, Telegram will start playing videos and music right away, without waiting for the files to fully download.
Calls
Use Less Data for Calls
Only while roaming
Respond with Text
Proxy
Proxy Settings

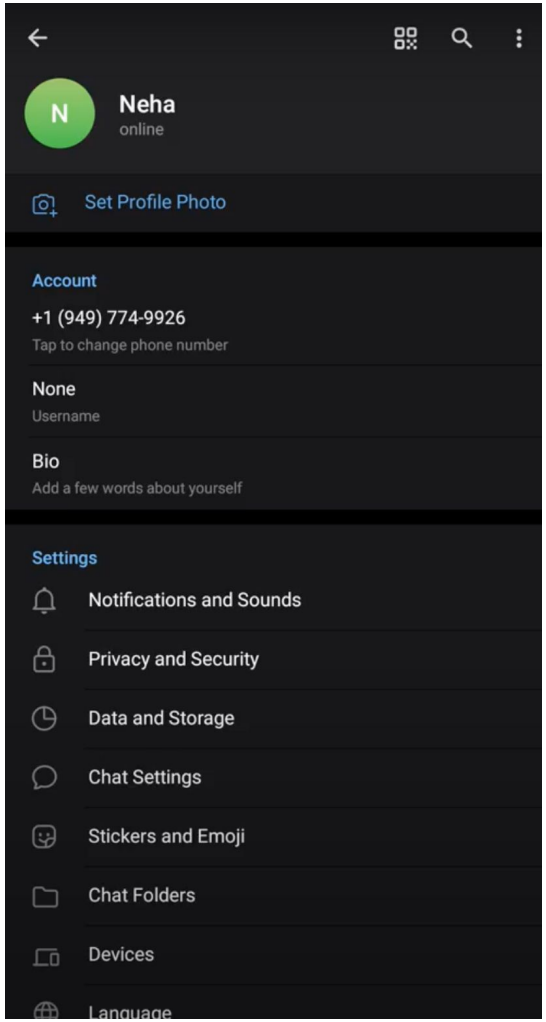

Language
Google may have access to the messages you translate.
Language
English
English
العربية
Arabic
Беларуская
Belarusian
Català
Catalan
Hrvatski
Croatian
Čeština
Czech
Nederlands
Dutch
Suomi
Finnish
Français
French
Deutsch
German
Magyar
Hungarian

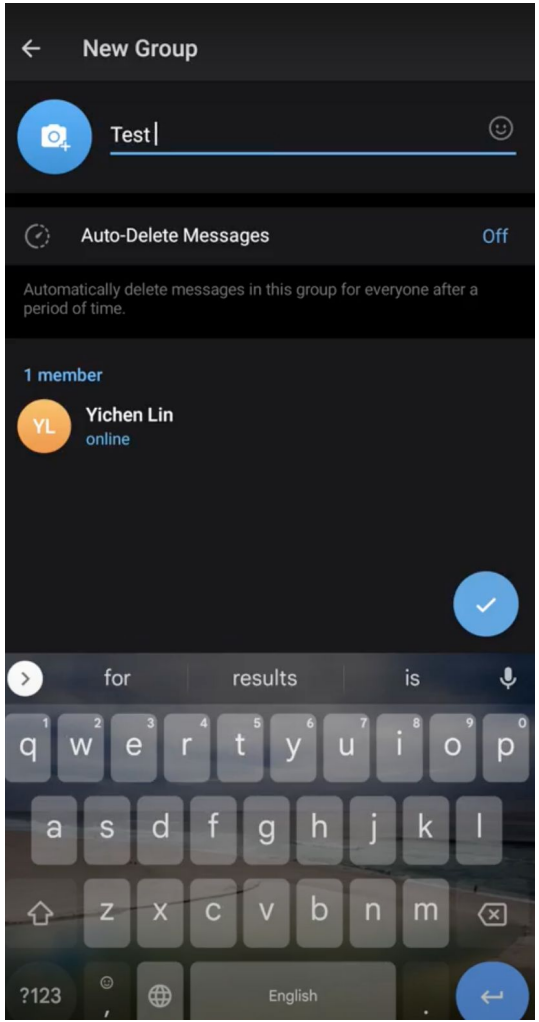

Telegram
TE
Test for 264P
12:03 PM
You: Hello
YL
Yichen Lin
12:02 PM
Hi
1
Telegram
12:01 PM
Login code: 27292. Do not give this code to a...
2
ZR
Zol. Ras.Mehrabani
Thu
Sticker
1
Your contacts on Telegram
Sachin Kadam
last seen at 10:46 AM
HK
Hardik C KJ
last seen at 10:03 AM
AM
Ashna's Mom
last seen at 9:35 AM
SM
Shubhankar Mt
last seen at 9:15 AM
Hash
last seen at 9:02 AM
DC
Dhiti CSI
last seen at 8:38 AM
Y
Yash
last seen at 8:06 AM
Arpit Sharma US
last seen at 6:51 AM


YL
Yichen Lin
online
Mute
Video Call
Search
Clear History
Change colors
Delete chat
February 19
Hello 12:02 PM
Hi 12:02 PM


YL
Yichen Lin
online
Info
+1 (949) 695-1914
Mobile
Notifications
On
Groups
TE
Test for 264P
2 members


N
Neha
+1 (949) 774-9926
Neha
Add Account
New Group
Contacts
Calls
People Nearby
Saved Messages
Settings
Invite Friends
Telegram Features


N
Neha
online
Set Profile Photo
Account
+1 (949) 774-9926
Tap to change phone number
None
Username
Bio
Add a few words about yourself
Settings
Notifications and Sounds
Privacy and Security
Data and Storage
Chat Settings
Stickers and Emoji
Chat Folders
Devices
Language


New Group
Test
Auto-Delete Messages
Off
Automatically delete messages in this group for everyone after a period of time.
1 member
YL
Yichen Lin
online
for
results
is
?123
English