



Sultan Qaboos University

College of Science

Department of computer science

WhatsApp

Final Report

Team members:

Salim Salah, id: 117710

Yahya Al Saidi, id: 113703

Contents

Introduction	3
Description.....	3
Targeted Audience	3
Tools	3
Requirement Specifications	4
Non-Functional Requirements	4
Functional Requirements.....	4
Design phase	5
Detailed Class Diagram	6
User Interface.....	10
Login interface design.....	10
Verification interface design	10
Main screen interface design	11
Find users screen interface design	11
Edit profile interface design	12
Chat with someone interface design	12
Implementation Phase	13
Class Hierarchy	13
Testing Phase	15
Difficulties	15
Future Work	16
Conclusion.....	16

Introduction

People are spending more time on mobile applications, especially chatting apps. Hundreds of chatting apps are available on google store and are the most dominating apps in terms of number of users. In this report, we are going to go through the process of developing out application starting from a brief description about the app, then requirements specification and design analysis, interface design, implementation, and testing.

Description

Our project is about an application that is used for direct messaging. The application provides you a list of all the contact that have downloaded the app along with their name and profile picture and you have the ability to choose any contact or a group of contacting and open a room for chatting. It has some additional functionalities which will be mentioned in the requirement specifications.

Targeted Audience

Android mobile users

Tools

We used the following tools to make this app:

- 1- Android studio: IDE for android applications development.
- 2- Firebase: is a Backend-as-a-Service (BaaS).
- 3- OneSignal: is a high volume and reliable push notification service for websites and mobile applications.
- 4- Iconsdp: is a website has a free collection of icons.
- 5- Materialpalette: is a website used to generate a palette for any color you input for the UI design.

Requirement Specifications

As we are the stakeholder of the applications, we have discussed about the functional and nonfunctional requirement the app should satisfy.

Non-Functional Requirements

- 1- Security: users' information and messages are secure.
- 2- Good performance: the app functionate efficiently in the worst cases of real time database usages.
- 3- Availability: the app should be available for all user around the world for free.

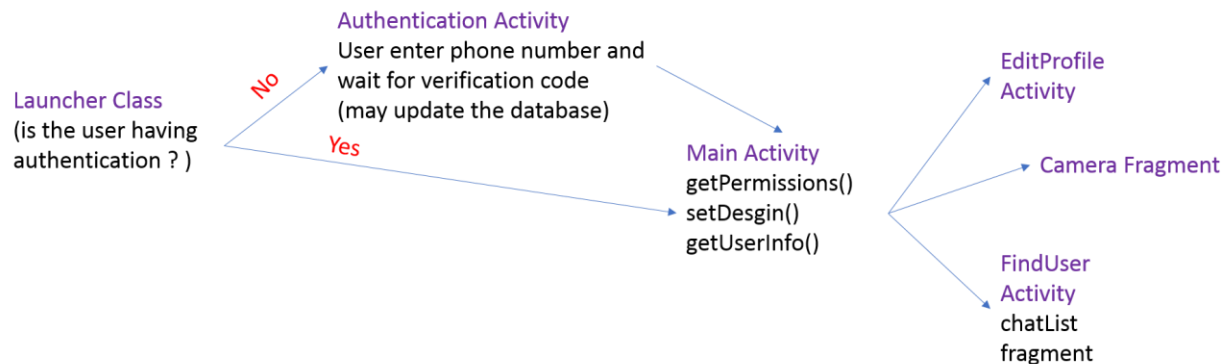
Functional Requirements

With our app you can:

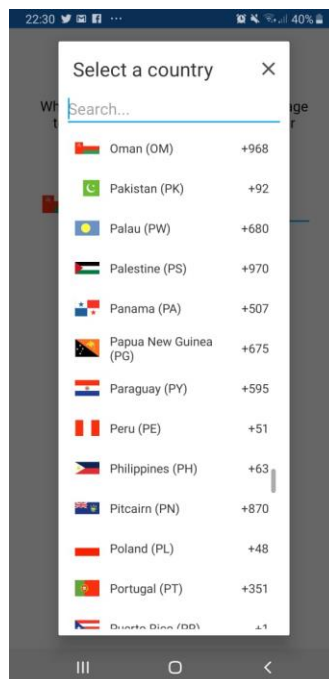
- Send a direct message to a user (one to one message).
- Emoji support.
- Attach pictures support.
- Create group chat.
- View the messages history.
- The read receipts feature allows your users to see when a message has been sent, delivered and read.
- Conversations list view with the last messages sent.
- View the user profile with full name and phone number.
- Login with phone number (Use phone number and code authentication method).
- Signup with phone number, verification code.
- Contacts list view.
- Logout from account.

Design phase

After we set the functional and nonfunctional requirements of the app, we searched a lot about different designs of chatting apps, and we come up with this initial general flow of control of the app activities:



We have used a lot of built in classes/packages and a ready programmer defined package (com.hbb20.CountryCodePicker): Which is responsible for the design of the main component in the Authentication Activity.



The component from com.hbb20.CountryCodePicker package.

Detailed Class Diagram

After knowing the basic components, objects, fragments, and interfaces of the app, We have designed this detailed Class Diagram.



```

+ ChatListAdapter extends RecyclerView.Adapter
┌ fields
~ chatList: ArrayList<ChatObject>
~ userList: ArrayList<UserObject>
~ context: Context
~ mainTab: boolean
└ constructors
+ ChatListAdapter( context: Context , chatList: ArrayList<ChatObject> , userList: ArrayList<UserObject> , mainTab: boolean )
┌ methods
+ onCreateViewHolder( parent: ViewGroup , viewType: int ): ChatListViewHolder
+ onBindViewHolder( holder: ChatListViewHolder , position: int ): void
+ getItemCount(): int

```

```

+ MessageAdapter extends RecyclerView.Adapter
┌ fields
~ messageList: ArrayList<MessageObject>
~ chatObject: ChatObject
~ context: Context
└ constructors
+ MessageAdapter( context: Context , chatObject: ChatObject , messageList: ArrayList<MessageObject> )
┌ methods
+ onCreateViewHolder( parent: ViewGroup , viewType: int ): MessageViewHolder
+ onBindViewHolder( holder: MessageViewHolder , position: int ): void
+ getItemCount(): int

```

```

+ DisplayImageFrame... extends Fragment
implements View.OnClickListener
┌ fields
~ view: View
~ mImage: ImageView
~ mSend: ImageButton
└ constructors
┌ methods
+ newInstance(): DisplayImageFragment
+ onCreateView( inflater: LayoutInflater , container: ViewGroup , savedInstanceState: Bundle ): View
~ initializeObjects(): void
+ onClick( v: View ): void

```

```

+ SendMessage
┌ fields
~ activity: Activity
~ totalMediaUploaded: int
~ mediaUriList: ArrayList<String>
~ mediaIdList: ArrayList<String>
~ chatObject: ChatObject
~ bitmap: Bitmap
~ fromMessageActivity: Boolean
~ mChatMessagesDb: DatabaseReference
~ mChatInfoDb: DatabaseReference
└ constructors
+ SendMessage( chatObject: ChatObject , fromMessageActivity: Boolean , mediaUriList: ArrayList<String> , bitmap: Bitmap , message: String )
┌ methods
~ updateDatabaseWithNewMessage( newMessageDb: DatabaseReference , newMessageMap: Map ): void

```

```

+ LauncherActivity extends AppCompatActivity
┌ fields
~ mAuth: FirebaseAuth
└ constructors
┌ methods
# onCreate( savedInstanceState: Bundle ): void

```

```

+ final MyAppGlideModul.. extends AppGlideModule
┌ fields
└ constructors
┌ methods

```

```

+ CodeFrame... extends Fragment
┌ fields
~ view: View
~ mCode: PinView
~ mPhone: TextView
~ mPhoneLong: TextView
└ constructors
┌ methods
+ onCreate( savedInstanceState: Bundle ): void
+ onCreateView( inflater: LayoutInflater , container: ViewGroup , savedInstanceState: Bundle ): View
+ onCreateViewCreated( view: View , savedInstanceState: Bundle ): void
~ codeInputHandling(): void
~ setPhoneText(): void
~ initializeObjects(): void

```

```

+ CameraViewFragm... extends Fragment
implements View.OnClickListener
┌ fields
~ view: View
~ mCamera: CameraView
~ mReverse: ImageButton
~ mProfile: ImageButton
~ mFlash: ImageButton
~ mCapture: ImageButton
└ constructors
┌ methods
+ newInstance(): CameraViewFragment
+ onCreateView( inflater: LayoutInflater , container: ViewGroup , savedInstanceState: Bundle ): View
~ initializeObjects(): void
+ captureImage(): void
~ reverseCameraFacing(): void
~ flashClick(): void
+ onClick( v: View ): void
+ onResume(): void
+ onPause(): void

```



```

+ UserObject
    implements Serializable
        Comparable
fields
- uid : String
- name : String
- phone : String
- image : String
- status : String
- notificationKey : String
- selected : Boolean
constructors
+ UserObject ( uid : String )
+ UserObject ( uid : String , name : String , phone : String , image : String , status : String )
methods
+ getUid () : String
+ getPhone () : String
+ getName () : String
+ getNotificationKey () : String
+ getImage () : String
+ getSelected () : Boolean
+ getStatus () : String
+ setName ( name : String ) : void
+ setNotificationKey ( notificationKey : String ) : void
+ setSelected ( selected : Boolean ) : void
+ setImage ( image : String ) : void
+ setStatus ( status : String ) : void
+ setPhone ( phone : String ) : void
+ compareTo ( o : Object ) : Int
+ equals ( o : Object ) : boolean

```

```

+ ChatListFragment extends Fragment
fields
- mChatList : RecyclerView
- mChatListAdapter : Adapter
- mChatListLayoutManager : LayoutManager
- chatList : ArrayList<ChatObject>
- userList : ArrayList<UserObject>
- view : View
- started : Boolean
- mainTab : boolean
- mAppBar : AppBarLayout
constructors
+ ChatListFragment ()
methods
+ newInstance ( mainTab : boolean ) : ChatListFragment
+ onCreate ( savedInstanceState : Bundle ) : void
+ onCreateView ( inflater : LayoutInflater , container : ViewGroup , savedInstanceState : Bundle ) : View
+ onViewCreated ( view : View , savedInstanceState : Bundle ) : void
+ getUserChatList () : void
+ getChatData ( chatId : String ) : void
+ getUserData ( mUser : UserObject ) : void
+ updatePaddingTop () : void
+ initializeRecyclerView () : void

```

```

+ UserListAdapter extends RecyclerView.Adapter
fields
- context : Context
- userList : ArrayList<UserObject>
- selectingMultipleUsers : boolean
constructors
+ UserListAdapter ( context : Context , userList : ArrayList<UserObject> )
methods
+ onCreateViewHolder ( parent : ViewGroup , viewType : int ) : UserListViewHolder
+ onBindViewHolder ( holder : UserListViewHolder , position : int ) : void
+ updateUserClick ( holder : UserListViewHolder ) : void
+ updateSelectingMultipleUsers () : void
+ getItemCount () : int

```

```

+ MessageObject
fields
- messageId : String
- senderId : String
- message : String
- timestampStr : String
- timestamp : Long
- mediaUriList : ArrayList<String>
constructors
+ MessageObject ( messageId : String , senderId : String , message : String , timestamp : Long , mediaUriList : ArrayList<String> )
methods
+ getMessageId () : String
+ getSenderId () : String
+ getMessage () : String
+ getTimestamp () : Long
+ getTimestampStr () : String
+ getMediaUriList () : ArrayList<String>
+ getDate () : void

```

```

+ CountryToPhonePref...
fields
- country2phone : Map<String , String>
constructors
methods
+ getPhone ( code : String ) : String
+ getAll () : Map<String , String>

```

```

+ ChatActivity extends AppCompatActivity
fields
- mChat : RecyclerView
- mMedia : RecyclerView
- mChatAdapter : Adapter
- mMediaAdapter : Adapter
- mChatLayoutManager : LayoutManager
- mMediaLayoutManager : LayoutManager
- messageList : ArrayList<MessageObject>
- mChatObject : ChatObject
- mChatMessagesDb : DatabaseReference
- mChatInfoDb : DatabaseReference
- PICK_IMAGE_INTENT : int
- mediaUriList : ArrayList<String>
constructors
methods
# onCreate ( savedInstanceState : Bundle ) : void
- getChatInfo () : void
- getChatMessages () : void
- sendMessage () : void
- initializeMessage () : void
- updateChatInfoViews () : void
- initializeMedia () : void
- openConfig () : void
- openGallery () : void
# onActivityResult ( requestCode : int , resultCode : int , data : Intent ) : void

```

```

+ ChatConfigActivity extends AppCompatActivity
    implements View.OnClickListener
fields
- mBack : ImageView
- mConfirm : ImageView
- mName : EditText
- mImage : ImageView
- mAuth : FirebaseAuth
- mInfoDatabase : DatabaseReference
- userId : String
- name : String
- image : String
- resultUri : Uri
- chatObject : ChatObject
- mDialog : ProgressDialog
constructors
methods
# onCreate ( savedInstanceState : Bundle ) : void
+ showProgressDialog ( message : String ) : void
+ dismissProgressDialog () : void
- getChatInfo () : void
- saveChatInformation () : void
+ onActivityResult ( requestCode : int , resultCode : int , data : Intent ) : void
+ onClick ( view : View ) : void
- initializeObjects () : void

```


+ ProfileEditFrage... extends `Fragment`
implements `View.OnClickListener`

fields

```
~ mBack: ImageView
~ mConfirm: ImageView
~ mName: EditText
~ mStatus: EditText
~ mPhone: EditText
~ mProfileImage: ImageView
~ mAuth: FirebaseAuth
~ mUserDatabase: DatabaseReference
~ userId: String
~ name: String
~ image: String
~ status: String
~ phone: String
~ resultUri: Uri
```

constructors

```
+ newInstance(): ProfileEditFragment
+ onCreateView( inflater: LayoutInflater, container: ViewGroup, savedInstanceState: Bundle ): View
+ getUserInfo(): void
+ saveUserInformation(): void
+ Logout(): void
+ onActivityResult( requestCode: int, resultCode: int, data: Intent ): void
+ onClick( view: View ): void
+ initializeObjects( view: View ): void
```

+ ChatObject

implements `Serializable`
`Comparable`

fields

```
- chatId: String
- lastMessage: String
- timestampStr: String
- image: String
- nameByUsers: String
- name: String
~ timestamp: Long
- userObjectArrayList: ArrayList<UserObject>
```

constructors

+ `ChatObject(chatId: String)`

methods

```
+ getChatId(): String
+ setName( name: String ): void
+ setNameByUsers( nameByUsers: String ): void
+ setLastMessage( lastMessage: String ): void
+ setTimestamp( timestamp: Long ): void
+ setImage( image: String ): void
+ getUserObjectArrayList(): ArrayList<UserObject>
+ getName(): String
+ getNameByUsers(): String
+ getLastMessage(): String
+ getTimestamp(): Long
+ getTimestampStr(): String
+ getImage(): String
+ addUserToArrayList( mUser: UserObject ): void
+ getDate(): void
+ compareTo( f: ChatObject ): int
```

+ AuthenticationActivity extends `AppCompatActivity`

fields

```
~ fm: FragmentManager
~ phoneFragment: PhoneFragment
~ mCallbacks: OnVerificationStateChangedCallbacks
~ phoneNumber: String
~ mVerificationId: String
~ dialog: ProgressDialog
```

constructors

methods

```
# onCreate( savedInstanceState: Bundle ): void
+ nextClick(): void
+ verifyPhoneNumberWithCode( code: String ): void
+ signInWithPhoneAuthCredential( phoneAuthCredential: PhoneAuthCredential ): void
+ startPhoneNumberVerification( phoneNumber: String ): void
+ getPhoneNumber(): String
+ usersLoggedIn(): void
+ onBackPressed(): void
```

+ MediaAdapter extends `RecyclerView.Adapter`

fields

```
~ mediaList: ArrayList<String>
~ context: Context
```

constructors

+ `MediaAdapter(context: Context, mediaList: ArrayList<String>)`

methods

```
+ onCreateViewHolder( parent: ViewGroup, viewType: int ): MediaViewHolder
+ onBindViewHolder( holder: MediaViewHolder, position: int ): void
+ getItemCount(): int
```

+ FindUserActivity extends `AppCompatActivity`

fields

```
- userList: RecyclerView
- userListAdapter: Adapter
- userListLayoutManager: LayoutManager
~ userList: ArrayList<UserObject>
~ contactList: ArrayList<UserObject>
```

constructors

methods

```
# onCreate( savedInstanceState: Bundle ): void
+ createChat(): void
+ getContactList(): void
+ getUserDetails( mContact: UserObject ): void
+ getCountryISO(): String
+ initializeRecyclerView(): void
```

+ PhoneFragment extends `Fragment`

implements `View.OnClickListener`

fields

```
~ mPhoneNumber: EditText
~ mNext: Button
~ ccp: CountryCodePicker
~ view: View
```

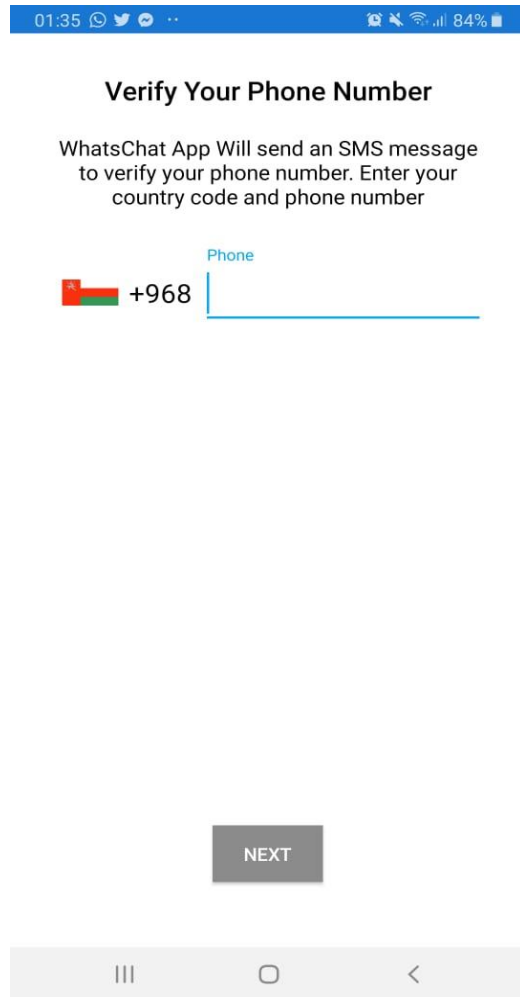
constructors

methods

```
+ onCreate( savedInstanceState: Bundle ): void
+ onCreateView( inflater: LayoutInflater, container: ViewGroup, savedInstanceState: Bundle ): View
+ onViewCreated( view: View, savedInstanceState: Bundle ): void
+ phoneEditTextButtonColor(): void
+ buildPhoneNumber(): String
+ onClick( view: View ): void
+ initializeObjects(): void
```

User Interface


Login interface design



01:35

Verify Your Phone Number

WhatsApp App Will send an SMS message to verify your phone number. Enter your country code and phone number

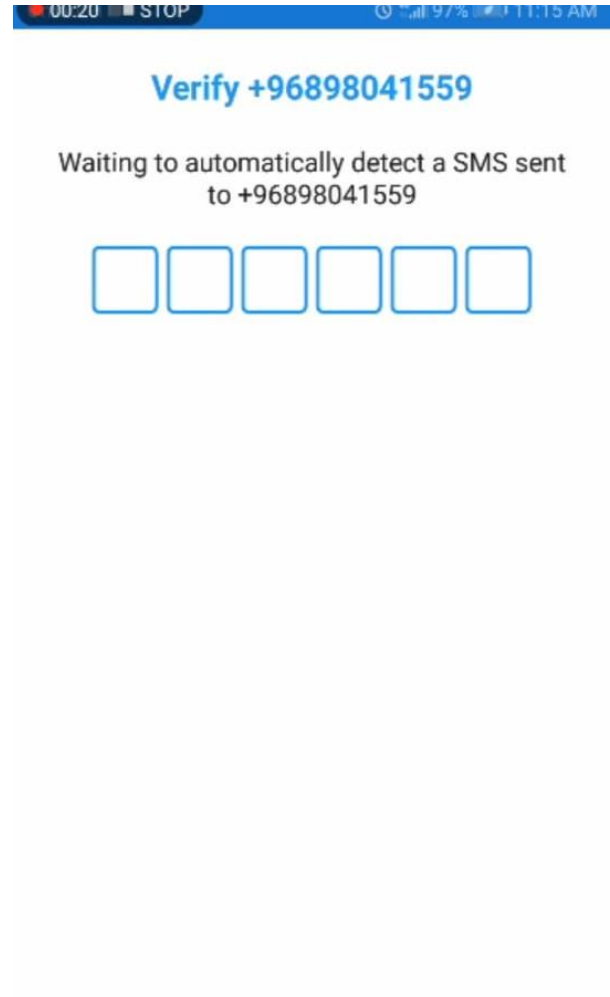
 +968

Phone

NEXT

III ○ <

Verification interface design



00:20 STOP

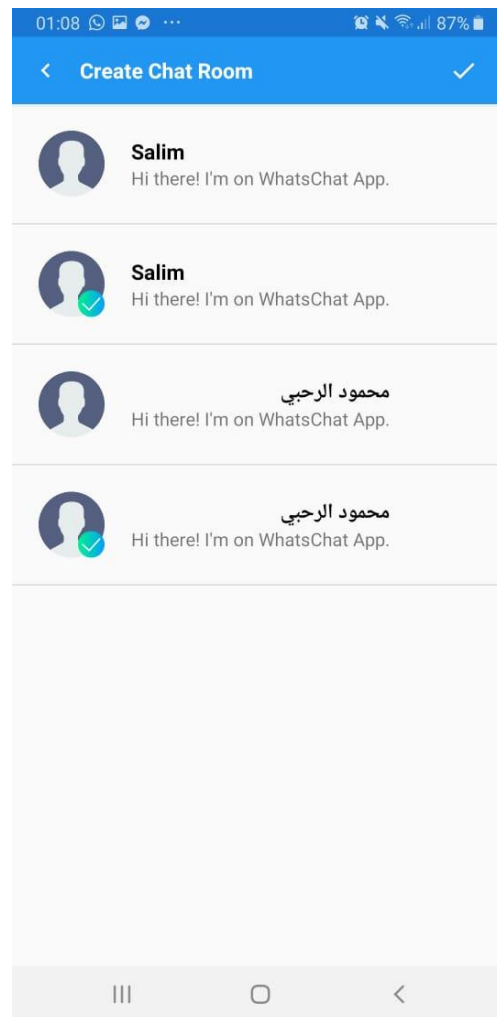
Verify +96898041559

Waiting to automatically detect a SMS sent to +96898041559

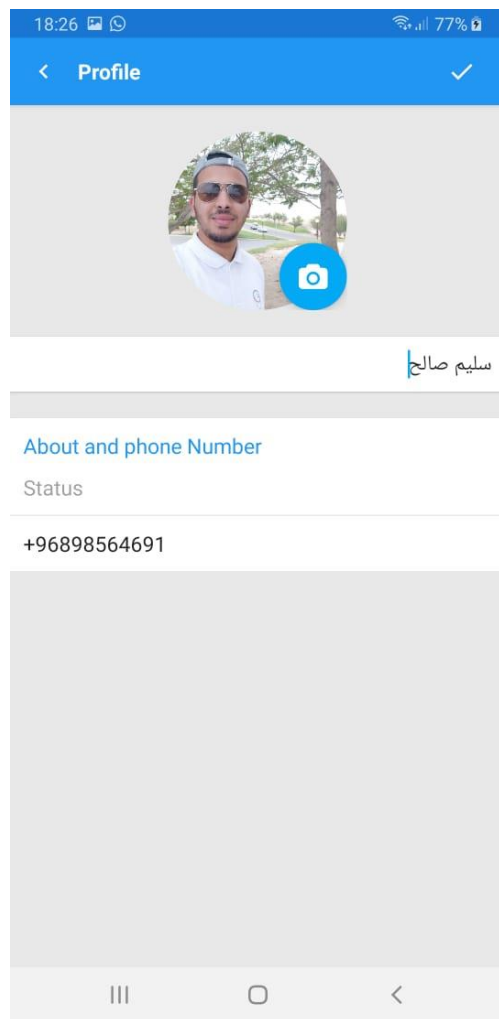
Main screen interface design



Find users screen interface design



Edit profile interface design



Chat with someone interface design



Implementation Phase

Class Hierarchy

- java.lang.Object
 - com.salimyahyaapp.whatschat.Object.**ChatObject** (implements java.lang.Comparable<T>, java.io.Serializable)
 - android.content.Context
 - android.content.ContextWrapper
 - android.view.ContextThemeWrapper
 - android.app.Activity (implements android.content.ComponentCallbacks2, android.view.KeyEvent.Callback, android.view.LayoutInflater.Factory2, android.view.View.OnCreateContextMenuListener, android.view.Window.Callback)
 - android.support.v4.app.SupportActivity (implements android.arch.lifecycle.LifecycleOwner)
 - android.support.v4.app.FragmentActivity (implements android.support.v4.app.ActivityCompat.OnRequestPermissionsResultCallback, android.support.v4.app.ActivityCompat.RequestPermissionsRequestCodeValidator, android.arch.lifecycle.ViewModelStoreOwner)
 - android.support.v7.app.AppCompatActivity (implements android.support.v7.app.ActionBarDrawerToggle.DelegateProvider, android.support.v7.app.AppCompatActivity.Callback, android.support.v4.app.TaskStackBuilder.SupportParentable)
 - com.salimyahyaapp.whatschat.Login.**AuthenticationActivity**
 - com.salimyahyaapp.whatschat.Chat.**ChatActivity**
 - com.salimyahyaapp.whatschat.Chat.**ChatConfigActivity** (implements android.view.View.OnClickListener)
 - com.salimyahyaapp.whatschat.**EditProfileActivity** (implements android.view.View.OnClickListener)
 - com.salimyahyaapp.whatschat.**FindUserActivity**
 - com.salimyahyaapp.whatschat.Login.**LauncherActivity**
 - com.salimyahyaapp.whatschat.**MainActivity**

- com.salimyahyaapp.whatschat.Utills.**CountryToPhonePrefix**
- com.salimyahyaapp.whatschat.**ExampleInstrumentedTest**
- com.salimyahyaapp.whatschat.**ExampleUnitTest**
- android.support.v4.app.Fragment (implements android.content.ComponentCallbacks, android.arch.lifecycle.LifecycleOwner, android.view.View.OnCreateContextMenuListener, android.arch.lifecycle.ViewModelStoreOwner)
 - com.salimyahyaapp.whatschat.Fragment.Camera.**CameraViewFragment** (implements android.view.View.OnClickListener)
 - com.salimyahyaapp.whatschat.Fragment.**ChatListFragment**
 - com.salimyahyaapp.whatschat.Login.**CodeFragment**
 - com.salimyahyaapp.whatschat.Fragment.Camera.**DisplayImageFragment** (implements android.view.View.OnClickListener)
 - com.salimyahyaapp.whatschat.Login.**PhoneFragment** (implements android.view.View.OnClickListener)
 - com.salimyahyaapp.whatschat.Fragment.**ProfileEditFragment** (implements android.view.View.OnClickListener)
- com.bumptech.glide.module.LibraryGlideModule
 - com.bumptech.glide.module.AppGlideModule
 - com.salimyahyaapp.whatschat.Utills.**MyAppGlideModule**
- com.salimyahyaapp.whatschat.Object.**MessageObject**
- android.support.v4.view.PagerAdapter
 - android.support.v4.app.FragmentPagerAdapter
 - com.salimyahyaapp.whatschat.**MainActivity.SectionsPagerAdapter**
- android.support.v7.widget.RecyclerView.Adapter<VH>
 - com.salimyahyaapp.whatschat.Adapter.**ChatListAdapter**
 - com.salimyahyaapp.whatschat.Adapter.**MediaAdapter**
 - com.salimyahyaapp.whatschat.Adapter.**MessageAdapter**
 - com.salimyahyaapp.whatschat.Adapter.**UserListAdapter**
- android.support.v7.widget.RecyclerView.ViewHolder
 - com.salimyahyaapp.whatschat.Adapter.**MediaAdapter.MediaViewHolder**
- com.salimyahyaapp.whatschat.Utills.**SendMessage**
- com.salimyahyaapp.whatschat.Utills.**SendNotification**
- com.salimyahyaapp.whatschat.Object.**UserObject** (implements java.lang.Comparable<T>, java.io.Serializable)

In additional to:

Firebase storage, authentication, and real time database set up.

OneSignal set up for notifications.

Testing Phase

The following is the acceptance testing:

| Test case name: WhatsChat App testing | | Reference: | |
|---|-----------------|---|--|
| Test case version: 1 | Sheet: 1 | Date created: 29/04/2019 | |
| Task or requirement: | | Notes: | |
| Send a direct message to a user (one to one message) | | Slow and sometimes stuck. | |
| Emoji support | | Done successfully | |
| Attach pictures support | | Slow in uploading the picture especially with high quality pictures | |
| Create group chat | | Done successfully | |
| View the messages history | | Done successfully | |
| The read receipts. | | Done successfully | |
| Conversations list view with the last messages sent | | Done successfully | |
| View the user profile with his/her name and phone number. | | Done successfully | |
| Login with phone number (Use phone number and code authentication method) | | Done successfully | |
| Signup with phone number, verification code | | Very fast authentication process | |
| Contacts list view. | | Done successfully | |
| Logout from account. | | Done successfully | |
| Notifications | | Sometime the system does not show them | |

Difficulties

1. the app dose not save the chat when the user is offline; because we need to some services to save the media and the chat from the Firebase.
2. The app was hard to do in a short time, because we learn each time something new and when we want to apply facing some problem so it take time to solve it and think of it.
3. Add some feature and delete feature take long time.
4. Hard to get some services from the Firebase because it need money to the commercial use to the app and not the normal use.

Future Work

- 1- improve the app by adding more feature like voice chat, SMS chat , send audio , and share moment with friends.
- 2- Change the service by using the Contus Fly services instead of google Firebase because it is more useful and provide more feature and more speed.
- 3- Improve the app speed by decreasing the number of codes that might be unnecessarily and put it to the cloud to handle it to make it faster like login page.
- 4- Adding end-to-end encryption to make the chat more privet and more secure so the user have the confident to use the app with respect to some aspect.
- 5- Adding the app as an web application and desktop application so the user can work and text from the phone or on his computer.

Conclusion

The app that we make was an chat app that you can use it to text and send photo to your friend. We have learn a lot of this course and we improve our self to make an apps to a commercial use or to the normal use for the daily live for our needs which can help us since the phone we have know can do a lot of things can help us to complete our tasks. We have faced some difficult but with some thinking and some search we have found some of what we want and some of things to add to improve our skills and our app. At the end we have learn how we can build a app that satisfy our needs and how to make it more efficient with the help of the instructor and the course.