POKE HOSPITAL

SUHAIL PURKAR
BRYAN ROMERO

CS 443

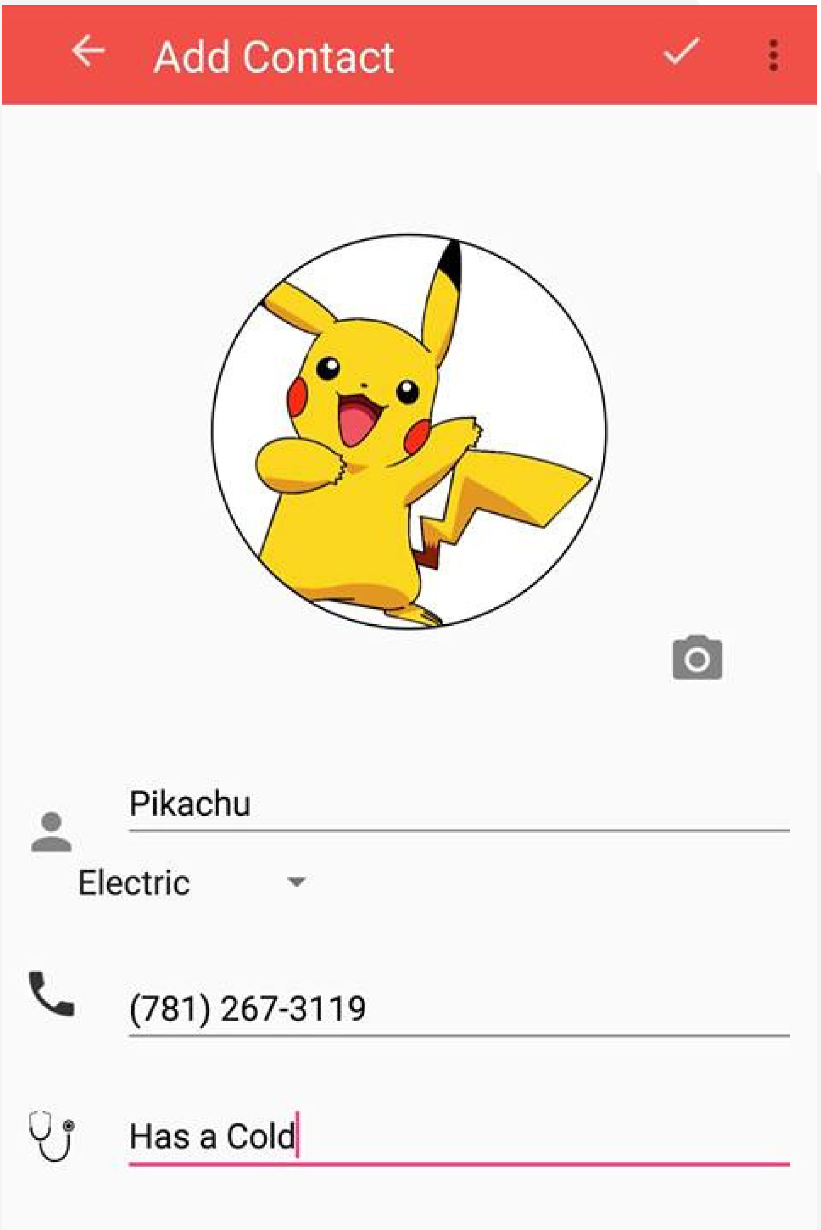
UNIVERSITY OF MASSACHUSETTS BOSTON

PROJECT STATEMENT

PROJECT PURPOSE

This application was developed as the Term project for Mobile Applications (CS 443) at Umass Boston. The project was chosen from the backup list of pre approved projects.

The Application is designed to help Nurses in the fictional Pokémon Universe. The application helps Nurses manage her patients and their medical history



Pikachu

Electric

(781) 267-3119

Has a Cold

The application interface allows nurses to add a contact for each Pokémon that visits the hospital

A contact consists of a picture selected by nurse from her camera or local storage, the phone number of the Pokémons trainer, and the diagnosis of the Pokémon on the given visit. The Nurse may later see a Pokémons contact information and medical history

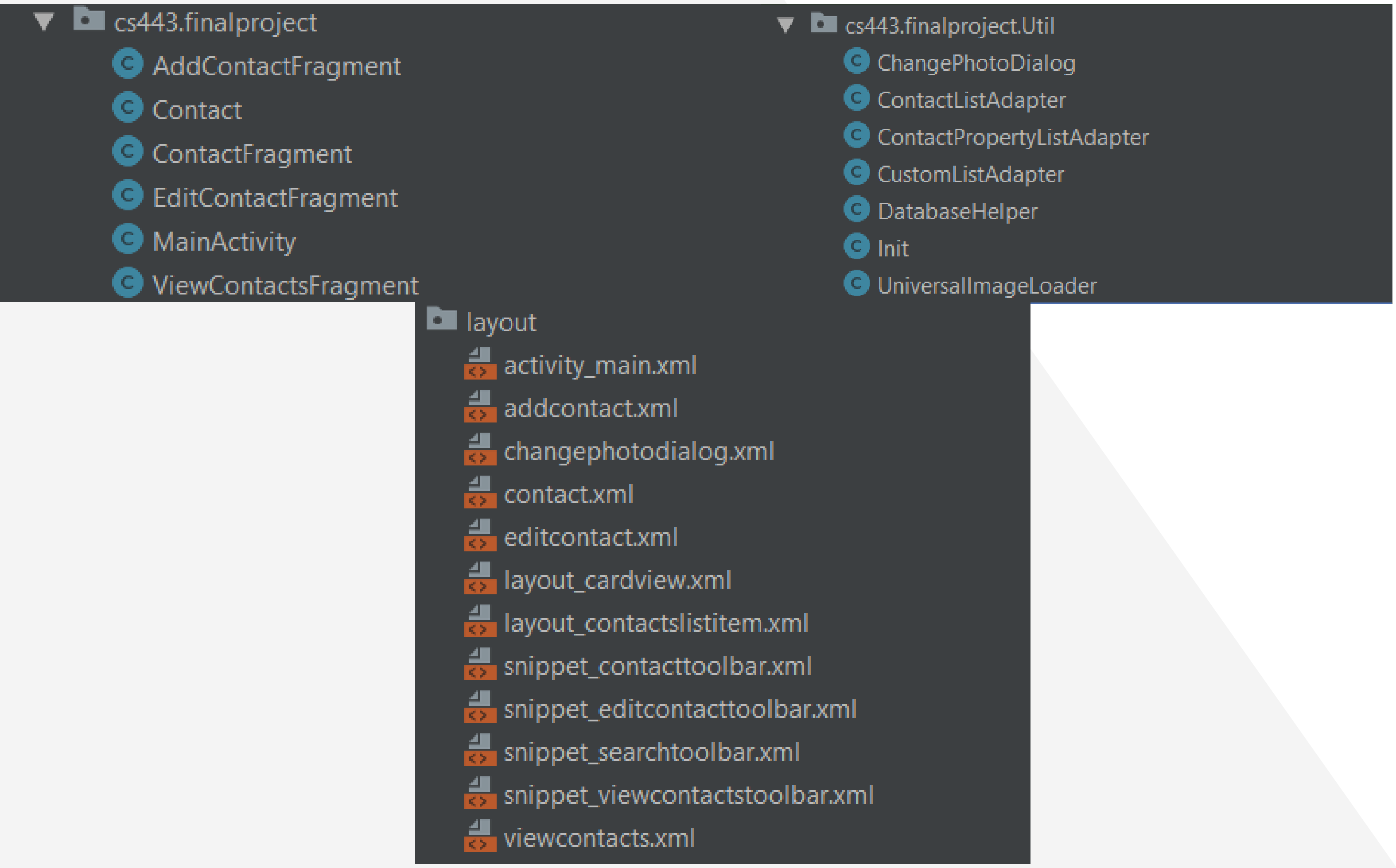
The functionality of the application is widely available on the App Store

APPLICATION DESIGN AND IMPLEMENTATION

DEVICE TARGET

The application is targeted for Android smartphone and tablet devices

MINSDKVERSION 18	Android Jellybean
TARGETSDKVERSION 26	Android Oreo



```
▼ cs443.finalproject
    AddContactFragment
    Contact
    ContactFragment
    EditContactFragment
    MainActivity
    ViewContactsFragment

▼ cs443.finalproject.Util
    ChangePhotoDialog
    ContactListAdapter
    ContactPropertyListAdapter
    CustomListAdapter
    DatabaseHelper
    Init
    UniversalImageLoader

layout
    activity_main.xml
    addcontact.xml
    changephotodialog.xml
    contact.xml
    editcontact.xml
    layout_cardview.xml
    layout_contactslistitem.xml
    snippet_contacttoolbar.xml
    snippet_editcontacttoolbar.xml
    snippet_searchtoolbar.xml
    snippet_viewcontactstoolbar.xml
    viewcontacts.xml
```

APPLICATION DESIGN AND IMPLEMENTATION

We used the class notes and first homework assignment to help create fragments for each individual page of the application

The application contains functionality to create, edit, delete and search for a contact

ADD

To add a contact a user clicks on the fabAddContact floating action button from the viewcontacts toolbar snippet. This navigates the application to the add contact fragment, where they may put in a contact image, type, name, phone number and diagnosis. To select a photo for the contact the user clicks on the ivCamera imageView which prompts a user for permissions to take a photo or access local storage, if the user accepts they are naviagted to the changephotodialog where they choose which option to proceed with .To save the contact the user clicks on the checkmark ImageView in the top right hand corner of the screen (editcontacttoolbar). This activates the confirmNewContact onClickListener in the AddContactFragment java file. We then create an instance of our Database helper and create an instance of the Contact object with the input received from the addcontact layout fragment. This Contact object is then added to our database using the helper instance we just created

APPLICATION DESIGN AND IMPLEMENTATION

EDIT

To Edit a contact a user clicks on the edit ImageView in the contacttoolbar snippet. This navigates the application to the edit contact fragment, where they may change the contact image, type, name, phone number and diagnosis. To save the contact the user clicks on the checkmark ImageView in the top right hand corner of the screen (editcontacttoolbar). This activates the   ivCheckMark onClickListener in the EditContactFragment java file. We then get a database helper instance and save the contact object

DELETE

To Delete a contact a user clicks on the delete menu item in the contact_menu. This activates the case R.id.menuitem_delete:
In the onOptionsItemSelected function of the EditContactFragment. A database helper instance is created and all arguments on the bundle are cleared and the contact object is deleted

SEARCH

To Search for a contact a user clicks on the ivSearchIcon imageview in the viewcontactstoolbar snippet. This activates the   ivSearchContact onClickListener in the ViewContactFragment java file. When the user clicks on the search bar we call toggleToolBarState() which Initiates the appbar state toggle. When the user types some input into the etSearchContacts edittext the addTextChanged listener in the viewcontactsfragment java file is activated and the contacts are filtered for display based on the given input

EXPERIENCES AND THOUGHTS

At the begging of the project we underestimated the complexity of working on an android application as a group, at times we were using different SDKs or working on different versions of the project are ran into several errors as a result of that. A lack of rigorous commenting also led to much confusion, we attempted to remedy this by adding comments retroactively but learned that its always best practice to add them at the appropriate time

We had originally intended to use a NoSQL Databse (Google Firebase) but had difficulty implementing the search functionality and could not find the appropriate documentation online to troubleshoot the problems. We then opted to using the Android SQLite database because it had more widely available documentation and tutorials. This did not come without its own drawbacks. For some unsolved reason on launch the first contact added does not save but every contact afterwards functions as designed. We also lost the functionality for displaying the contact photo, we had originally believed it to just be a problem with getting the insufficient permissions but ruled this out since it had been working earlier with FireBase. FireBase also had an extremely simple authentication interface while we couldn't quite figure out the implementation for SQlite without the application crashing repeatedly.

REFRENCES

We utilized the class notes and homeworks, releveant Android Studio, Firebase and SQlite documentation and tutorials from the web