Mobile Embedded Devices
Group Contribution Report
UFCFW5-30-2

Andrew Belcher
StudentID:17010347

March 28, 2019

The code discussed in this report regarding the ”Spy App” can be found at..

`https://gitlab.uwe.ac.uk/a2-belcher/med`

Contents

1 Task specification

1.1 Android App Development

This will involve you writing an Android app with a login interface and secure data storage (encrypted data) to store the username and password. Once the user logs in, the "Spy App" will allow recording of audio and the capture of still images using the built-in microphone and camera. Implementing basic camera features and audio capture will be achieved from the camera and audio worksheet. Additional functional requirements will require group self-directed study of the relevant topics.

1.2 Android OS customization

Following the provided worksheets, download/configure/and compile Lineage OS 14.1 source code(flo) for the Nexus 7 tablet. Demonstrate a custom boot animation by swapping out the corresponding files and rebuilding the OS so that it can be installed on the device. Detail any issues experienced.

2 Spy App

Tasked with developing a ”Spy App” for an Android Nexus 7 tablet under API 23, our group was able to complete all requested functionally requirements. These ranged from the ability to take pictures and record audio as well as store.

2.1 Develop Image Capture and Audio Recording Functionality

Following the previous image capture worksheet, i contributed my working camera app code to the ”Spy App” since it was pretty stable at this point. The code to do this is performed in a few functions..

```

    private android.hardware.Camera.PictureCallback captureMedia =
        new Camera.PictureCallback()
    {

        @Override
        public void onPictureTaken(byte[] data, Camera camera) {
            Matrix matrix = new Matrix();
            matrix.postRotate(90F);
            Bitmap bitmap = BitmapFactory.decodeByteArray(data, 0,
                data.length);

            Bitmap newBitmap = Bitmap.createBitmap(bitmap,0, 0,
                bitmap.getWidth(), bitmap.getHeight(),   matrix,
                true);

            if (bitmap == null) {
            } else {
                pictureTaken.setImageBitmap(newBitmap);
                pictureTaken.setBackgroundResource(R.drawable.
                    ic_launcher_background);
                pictureTaken.setZ(6);
                pictureTaken.setVisibility(View.VISIBLE);
                Button captureButton = findViewById(R.id.
                    captureButton);
                Button recordButton = findViewById(R.id.
                    recordButton);
                captureButton.setEnabled(false);
                recordButton.setEnabled(false);
                captureButton.setVisibility(View.INVISIBLE);
                recordButton.setVisibility(View.INVISIBLE);
                FrameLayout cameraPreview = findViewById(R.id.
                    camPreview);
                cameraPreview.setVisibility(View.INVISIBLE);
                ConstraintLayout background = findViewById(R.id.
                    camBackground);
                background.setBackgroundColor(Color.GRAY);

                // Make directory called RecordData if it doesn't
                    already exist
```

```java
                File f = new File(Environment.getExternalStorageDirectory(),
                        folder_main);
                if (!f.exists()) {
                    f.mkdirs();
                }
                String path = Environment.getExternalStorageDirectory().toString();

                // enter the following code statement all on one line
                String filename = path + "/" + folder_main + "/" + "recordData" + String.format("%d.jpg", System.currentTimeMillis());
                FileOutputStream outStream = null;
                try {
                    outStream = new FileOutputStream(filename);
                    outStream.write(data);
                    outStream.close();
                } catch (FileNotFoundException e) {
                    e.printStackTrace();
                } catch (IOException e) {
                    e.printStackTrace();
                } finally {
                }
            }
            cameraObject.startPreview();
        }
    };

    public void captureImage(View view)
    {
        cameraObject.takePicture(null, null, captureMedia);
    }

    public void recordAudio(View view)
    {
        if(!recording_running)
        {

            // Make directory called RecordData if it doesn't already exist
            File f = new File(Environment.getExternalStorageDirectory(), folder_main);

            if (!f.exists()) {
                f.mkdirs();
            }

            String path = Environment.getExternalStorageDirectory().toString();

            // enter the following code statement all on one line
            String filename = path + "/" + folder_main + "/" + "recordData" + String.format("%d.3gp", System.currentTimeMillis());
            recorder = new MediaRecorder();
            recorder.setAudioSource(MediaRecorder.AudioSource.MIC);
            recorder.setOutputFormat(MediaRecorder.OutputFormat.THREE_GPP);
            recorder.setAudioEncoder(MediaRecorder.AudioEncoder.AMR_NB);
            recorder.setOutputFile(filename);

            try {
                recorder.prepare();
            } catch (IllegalStateException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            } catch (IOException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
            recorder.start(); // Recording is now started
            recorder_open = true;
        }

        else
            recorder.stop();

        recording_running = !recording_running;
    }
```

This aspect of the app also required graceful permission requesting which was accomplished with the following..

```java


    private static final int REQUEST_CAMERA = 0;
    private static final int REQUEST_CONTACTS = 1;
    private static final int REQUEST_EXTERNAL_STORAGE = 2;
    private static final int AUDIO_RECORD_REQUEST_CODE = 300;
    private static String[] PERMISSIONS_CONTACT = {Manifest.
        permission.READ_CONTACTS, Manifest.permission.
        WRITE_CONTACTS};

    String[] PERMISSIONS = {
            android.Manifest.permission.WRITE_EXTERNAL_STORAGE,
            android.Manifest.permission.RECORD_AUDIO,
            android.Manifest.permission.CAMERA
    };

    // CLASS METHODS
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        cur_page = current_page.LOGIN;
        prev_page = current_page.LOGIN;

        setContentView(R.layout.activity_spy_app_login);
        initCrypto();

        if(checkAppPermissions())
            requestAppPermissions();

        enableTextListenier();
    }

...

    private void requestAppPermissions()
    {
        Log.i(TAG, "CAMERA/Record audio/write storage permission
            has NOT been granted. Requesting permission.");
        ActivityCompat.requestPermissions(SpyApp.this, PERMISSIONS,
             1);
    }

    private boolean checkAppPermissions()
    {
        return
                (
                // we dont have camera privs
                ActivityCompat.checkSelfPermission(this, Manifest.
                    permission.CAMERA)
                != PackageManager.PERMISSION_GRANTED

                // we dont have audio privs
                || ActivityCompat.checkSelfPermission(this,
                    Manifest.permission.RECORD_AUDIO)
                != PackageManager.PERMISSION_GRANTED

                // we dont have write storage privs
                || ActivityCompat.checkSelfPermission(this,
                    Manifest.permission.WRITE_EXTERNAL_STORAGE)
                != PackageManager.PERMISSION_GRANTED);
    }
```

2.2 Develop a Login UI

Now that Camera/Recording functionality was maintained it was time for us to develop a login interface. My work towards this goal was focused on layout switching to focus on displaying a login layout on class creation. As well as input verification involving the login fields and linking different login related buttons to other layouts. Below is the final code of the application that related to login verification.

```java
public void handleLoginAttempt(View view)
{
    Log.i(TAG, "ON LOGIN CLICK attempts:" + login_attempts);

    EditText userText = (EditText)findViewById(R.id.login_username)
        ;
    EditText passText = (EditText)findViewById(R.id.login_password)
        ;
    String username = userText.getText().toString();
    String password = passText.getText().toString();


    // check if user exists!
    // check if password matchs
    if(checkAppPermissions()) {
        requestAppPermissions();
        userText.setText("");
        passText.setText("");
    }
    else {
        Document doc = getXmlDoc();

        // our credentials file
        File file = new File(Environment.
            getExternalStorageDirectory(), folder_main+"/
            credentials/details.sxml");
        if (file.exists())
        {

            if (username.equals("") || password.equals("")) {
                if (username.equals(""))
                    Toast.makeText(getApplicationContext(), "
                        Username cant be empty", Toast.LENGTH_SHORT
                        ).show();
                else
                    Toast.makeText(getApplicationContext(), "
                        Password cant be empty", Toast.LENGTH_SHORT
                        ).show();

                userText.setText("");
                passText.setText("");

            }

            else if (usernameExists(username, doc) &&
                credentialsMatch(username, password, doc)) {
```

```java
                drawApp();
            }
            else {

                    TextView attempt_text = findViewById(R.id.attempts_view);
                    attempt_text.setVisibility(View.VISIBLE);
                    TextView attempt_num = findViewById(R.id.attempt_num);
                    attempt_num.setVisibility(View.VISIBLE);

                    login_attempts--;
                    attempt_num.setText(Integer.toString(login_attempts));

                    userText.setText("");
                    passText.setText("");

                    if (login_attempts == 0) {
                        attempt_num.setTextColor(Color.RED);
                        attempt_text.setTextColor(Color.RED);

                        Button btn = (Button) findViewById(R.id.login_confirm);
                        btn.setEnabled(false);
                    }
                }
        }
        else
        {
            if (username.equals("") || password.equals("")) {
                if (username.equals(""))
                    Toast.makeText(getApplicationContext(), "Username cant be empty", Toast.LENGTH_SHORT).show();
                else
                    Toast.makeText(getApplicationContext(), "Password cant be empty", Toast.LENGTH_SHORT).show();
            }
            else {

                Toast.makeText(getApplicationContext(), "User does not exist, please register!", Toast.LENGTH_SHORT).show();
            }
            userText.setText("");
            passText.setText("");
        }
    }
}
```

2.3 Develop a User Registration Login UI

With the ability to connect a login layout to the camera layout and verify input (without credential verification yet) we could now go on to developing a registration page. I was able to develop code to verify fields as well and later on actually update the credentials file.

```java

    public void handleRegisterAttempt(View view) {

        EditText userText = (EditText)findViewById(R.id.
            reg_username);
        EditText passText = (EditText)findViewById(R.id.
            passchg_newpass);
        EditText passConfirmText = (EditText)findViewById(R.id.
            passchg_newpass_confirm);
        String username = userText.getText().toString();
        String password = passText.getText().toString();
        String passwordConfirm = passConfirmText.getText().toString
            ();

        // gain access to filesystem write priv first
        if (checkAppPermissions()) {
            requestAppPermissions();
            userText.setText("");
            passText.setText("");
            passConfirmText.setText("");
        }

            // handle register attempt
        else
        {

            File filedir = new File(Environment.
                getExternalStorageDirectory(), folder_main+"/
                credentials");

            // our credentials file
            File file = new File(Environment.
                getExternalStorageDirectory(), folder_main+"/
                credentials/details.sxml");

            if (username.equals("") || password.equals("")) {

                if (username.equals(""))
                    Toast.makeText(getApplicationContext(), "
                        Username cant be empty", Toast.LENGTH_SHORT
                        ).show();
                else if(password.equals(""))
                    Toast.makeText(getApplicationContext(), "
                        Password cant be empty", Toast.LENGTH_SHORT
                        ).show();

                userText.setText("");
                passText.setText("");
                passConfirmText.setText("");
            }

            else if(password.equals(passwordConfirm)) {
                if (file.exists()) {

                    Document doc = getXmlDoc();

                    if (!usernameExists(username, doc))
                    {
                        addNewUser(username, password,doc);
                    }
                    else {
                        Toast.makeText(getApplicationContext(), "
                            name is already taken", Toast.
                            LENGTH_SHORT).show();
                    }
                }
                else {

                    filedir.mkdirs();

                    String templateCredData = "<?xml version
                        =\"1.0\" encoding=\"UTF-8\"?>\n\n" +
                            "<credentials>\n" +
                            "    <user id=" + '"' + "%d" + '"' + ">\
                                n" +
                            "        <name>%s</name>\n" +
                            "        <pass>%s</pass>\n" +
                            "    </user>\n" +
                            "</credentials>";

                    templateCredData = String.format(
                        templateCredData, 1, username, password);

                    if (writeCredentialsFile(templateCredData)) ;
                    else
                        Log.i(TAG, "Error in writing encrypted file
                             back!!");

                }
                onBackPressed();
            }
            else
            {
                userText.setText("");
                passText.setText("");
                passConfirmText.setText("");

                Toast.makeText(getApplicationContext(), "
                    Confirmation password does not match!", Toast.
                    LENGTH_SHORT).show();
            }

        }
    }
```

2.4 Store Encrypted Username and Password

To tie most of the needed login functionality together, we needed to store credentials on the file system which would also need to be encrypted for safe storage. My role was to develop code to parse an xml file that is initially encrypted in `AES-128-CBC`. This would require me to also implement code to encrypt/decrypt the file before storing and retrieving.

```

    // Encrypt a byte array in AES-128-CBC
    public byte[] encrypt(byte[] data)
    {
        try
        {
            byte[] iv = new byte[16];
            cipher.init(ENCRYPT_MODE, skey, new IvParameterSpec(iv)
                );
            DeflaterInputStream deflaterInput = new
                DeflaterInputStream(new CipherInputStream(new
                DeflaterInputStream(new ByteArrayInputStream(data))
                , cipher));
            Log.i(TAG,"iv:" + byteArrayToHex(iv));
            return IOUtils.toByteArray(deflaterInput);
        }

        catch (Exception e)
        {
            e.printStackTrace();
            return new byte[0];
        }
    }

    // Decrypt a byte array in AES-128-CBC
    public byte[] decrypt(byte[] data)
    {
        byte[] result = new byte[0];

        try
        {
            byte[] iv = new byte[16];
            cipher.init(DECRYPT_MODE, skey, new IvParameterSpec(iv)
                );

            InflaterInputStream inflaterStream = new
                InflaterInputStream(new CipherInputStream(new
                InflaterInputStream(new ByteArrayInputStream(data))
                , cipher));
            return IOUtils.toByteArray(inflaterStream);
        }
        catch (Exception e)
        {
            Log.e(TAG, "decryption out failed: "+Log.
                getStackTraceString(e));
            return result;
        }
    }

    // Pass in document file to convert to StringWriter for xml
        encryption
    public StringWriter writeXml(Document doc)
    {
        StringWriter writer = null;

        try
        {
            DOMSource domSource = new DOMSource(doc);
            writer = new StringWriter();
            StreamResult result = new StreamResult(writer);
            TransformerFactory tf = TransformerFactory.newInstance
                ();
            Transformer transformer = tf.newTransformer();
            transformer.transform(domSource, result);
            System.out.println("XML IN String format is: " + writer
                .toString());
        }

        catch (javax.xml.transform.
            TransformerConfigurationException e)
        {
            e.printStackTrace();
        }

        catch (javax.xml.transform.TransformerException e)
        {
            e.printStackTrace();
        }

        return writer;
    }

    public InputStream readCredentialsFile()
    {
        File file = new File(Environment.
            getExternalStorageDirectory(), folder_main+"/
            credentials/details.sxml");
        int size = (int) file.length();
        byte[] sxml = null;
        byte[] xml = null;

        sxml = new byte[size];

        try
        {
            BufferedInputStream buf = new BufferedInputStream(new
                FileInputStream(file));
            buf.read(sxml, 0, sxml.length);
            buf.close();
        }

        catch (FileNotFoundException e)
        {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
        catch (IOException e)
        {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }

        xml = decrypt(sxml);
        Log.i(TAG,"new decrypted data is: "+byteArrayToHex(xml));

        // String str = new String(dectest, "UTF-8"); // for UTF-8
            encoding
        InputStream decxml = new ByteArrayInputStream(xml);
        Log.i(TAG, "new decrypted file is: " + decxml);

        return decxml;
    }

    public boolean writeCredentialsFile(String credFile)
    {
        // Make directory called RecordData if it doesn't already
            exist
        //File f = new File(Environment.getExternalStorageDirectory
            (), folder_main+"/credentials/");
        String path = Environment.getExternalStorageDirectory().
            toString();
        String filename = path + "/" + folder_main + "/" + "
            credentials/" + "details.sxml";

        byte[] xml = null;
        byte[] sxml = null;

        try
        {
            Log.i(TAG,"key is: "+ byteArrayToHex(key));
            Log.i(TAG,"skey is: "+ byteArrayToHex(skey.getEncoded()
                ));
            Log.i(TAG,"key length: "+ key.length);

            xml = credFile.getBytes();
            Log.i(TAG,"decrypted data is: "+ byteArrayToHex(xml));
            sxml = encrypt(xml);
            Log.i(TAG,"encrypted data is: "+byteArrayToHex(sxml));
            xml = decrypt(sxml);
            Log.i(TAG,"new decrypted data is: "+byteArrayToHex(xml)
                );
            String str = new String(xml, "UTF-8"); // for UTF-8
                encoding
            Log.i(TAG,"new decrypted file is: "+str);
        }

        catch(java.lang.Exception e)
        {
            Log.e(TAG, "Exception: "+Log.getStackTraceString(e));
        }

        // enter the following code statement all on one line
        FileOutputStream outStream = null;

        try
        {
            outStream = new FileOutputStream(filename);
            outStream.write(sxml);
            outStream.close();
            return true;
        }
        catch (FileNotFoundException e)
        {
            e.printStackTrace();
        }
        catch (IOException e)
        {
            e.printStackTrace();
        }
        finally
        {
        }
        return false;
    }

    public Document getXmlDoc()
    {
        InputStream credFile = readCredentialsFile();

        Document doc = null;

        try{
            DocumentBuilderFactory dbFactory =
                DocumentBuilderFactory.newInstance();
            DocumentBuilder dBuilder = dbFactory.newDocumentBuilder
                ();
            doc = dBuilder.parse(credFile);

            Element element = doc.getDocumentElement();
            element.normalize();
        }

        catch(org.xml.sax.SAXException e)
        {
            Log.e(TAG, "sax exception failed: " + Log.
                getStackTraceString(e));
        }
        catch(javax.xml.parsers.ParserConfigurationException e)
        {
            e.printStackTrace();
        }
        catch(java.io.IOException e)
        {
            e.printStackTrace();
        }
        return doc;

    }
```

2.5 Verify Username and Password for Login Process

After implementing functionality needed to store credentials, we could move on to developing username and password verification to complete the login process. I provided code to look through the parsed credentials and determine if the input data matched with a stored users' credentials. First of which was to check if a user existed, and second was to determine if their credentials matched. Also, code for handling the login attempt was developed by me for verifying attempts and input data.

```


    public boolean usernameExists(String usernameChecked, Document doc)
    {
            Element element = doc.getDocumentElement();
            element.normalize();

            NodeList nList = doc.getElementsByTagName("user");

            for (int i = 0; i < nList.getLength(); i++) {

                Node node = nList.item(i);
                if (node.getNodeType() == Node.ELEMENT_NODE) {
                    if (node.getNodeName().equals("user")) {

                        Element element2 = (Element) node;
                        NodeList userCreds = element2.getChildNodes();

                        for (int d = 0; d < userCreds.getLength(); d++) {
                            Node credList = userCreds.item(d);

                            if (credList.getNodeType() == Node.ELEMENT_NODE) {
                                if (credList.getNodeName().equals("name")) {

                                    Element element3 = (Element) credList;

                                    if (usernameChecked.equals(element3.getTextContent())) {
                                        Log.i(TAG, "node: " + element3.getTextContent());
                                        return true;
                                    }
                                }
                            }
                        }
                    }
                }
            }
```

```
        return false;
    }

    public boolean credentialsMatch(String username, String
        password, Document doc)
    {
        Element element = doc.getDocumentElement();
        boolean onUserEntry = false;

        element.normalize();

        NodeList nList = doc.getElementsByTagName("user");

        for (int i = 0; i < nList.getLength(); i++) {

            Node node = nList.item(i);
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                if (node.getNodeName().equals("user")) {

                    Element element2 = (Element) node;
                    NodeList userCreds = element2.getChildNodes();

                    for (int d = 0; d < userCreds.getLength(); d++)
                         {
                        Node credList1 = userCreds.item(d);
                        //Node credList2 = userCreds.item(d+);

                        if (credList1.getNodeType() == Node.
                            ELEMENT_NODE) {
                            Element element3 = (Element) credList1;

                            if (credList1.getNodeName().equals("
                                name")) {
                                if (username.equals(element3.
                                    getTextContent())) {
                                    onUserEntry = true;
                                }
                            }
                            if(credList1.getNodeName().equals("pass
                                ") && onUserEntry)
                            {
                                if(password.equals(element3.
                                    getTextContent())){
                                    Log.i(TAG, "pass: " + element3.
                                        getTextContent());
                                    return true;
                                }
                                onUserEntry = false;
                            }
                        }
                    }
                }
            }
        }
        return false;

    }
```

```java

    public void handleLoginAttempt(View view)
    {
        Log.i(TAG, "ON LOGIN CLICK attempts:" + login_attempts);

        EditText userText = (EditText)findViewById(R.id.
            login_username);
        EditText passText = (EditText)findViewById(R.id.
            login_password);
        String username = userText.getText().toString();
        String password = passText.getText().toString();


        // check if user exists!
        // check if password matchs
        if(checkAppPermissions()) {
            requestAppPermissions();
            userText.setText("");
            passText.setText("");
        }
        else {
            Document doc = getXmlDoc();

            // our credentials file
            File file = new File(Environment.
                getExternalStorageDirectory(), folder_main+"/
                credentials/details.sxml");
            if (file.exists())
            {

                if (username.equals("") || password.equals("")) {
                    if (username.equals(""))
                        Toast.makeText(getApplicationContext(), "
                            Username cant be empty", Toast.
                            LENGTH_SHORT).show();
                    else
                        Toast.makeText(getApplicationContext(), "
                            Password cant be empty", Toast.
                            LENGTH_SHORT).show();

                    userText.setText("");
                    passText.setText("");

                }

                else if (usernameExists(username, doc) &&
                    credentialsMatch(username, password, doc)) {
                    drawApp();
                }
                else {

                        TextView attempt_text = findViewById(R.id.
                            attempts_view);
                        attempt_text.setVisibility(View.VISIBLE);
                        TextView attempt_num = findViewById(R.id.
                            attempt_num);
                        attempt_num.setVisibility(View.VISIBLE);

                        login_attempts--;
                        attempt_num.setText(Integer.toString(
                            login_attempts));

                        userText.setText("");
                        passText.setText("");

                        if (login_attempts == 0) {
                            attempt_num.setTextColor(Color.RED);
                            attempt_text.setTextColor(Color.RED);

                            Button btn = (Button) findViewById(R.id
                                .login_confirm);
                            btn.setEnabled(false);
                        }
                    }
            }
            else
            {
                if (username.equals("") || password.equals("")) {
                    if (username.equals(""))
                        Toast.makeText(getApplicationContext(), "
                            Username cant be empty", Toast.
                            LENGTH_SHORT).show();
                    else
                        Toast.makeText(getApplicationContext(), "
                            Password cant be empty", Toast.
                            LENGTH_SHORT).show();
                }
                else {

                    Toast.makeText(getApplicationContext(), "User
                        does not exist, please register!", Toast.
                        LENGTH_SHORT).show();
                }
                userText.setText("");
                passText.setText("");
            }
        }
    }
```

2.6 Develop a UI to Change the Password

At this point we had now had our app checking user credentials for logging into the camera section, as well as the ability to register a new user. But we hadn't implemented the functionality of changing an existing users' password. In this section of development, I was able to develop the code necessary to update the password after decrypting and parsing the credentials file, updating and restoring the file afterwards. Verification code for the password update pages fields were written by myself also.

```

    public void handlePassChangeAttempt(View view)
    {
        EditText userText = (EditText)findViewById(R.id.
            passchg_username);
        EditText oldPassText = (EditText)findViewById(R.id.
            passchg_prevpass);
        EditText newPassText = (EditText)findViewById(R.id.
            passchg_newpass);
        EditText newPassConfirmText = (EditText)findViewById(R.id.
            passchg_newpass_confirm);
        String username = userText.getText().toString();
        String oldPassword = oldPassText.getText().toString();
        String newPassword = newPassText.getText().toString();
        String newPasswordConfirm = newPassConfirmText.getText().
            toString();

        // gain access to filesystem write priv first
        if(checkAppPermissions()) {
            requestAppPermissions();

            userText.setText("");
            oldPassText.setText("");
            newPassText.setText("");
            newPassConfirmText.setText("");
        }
            // handle pass change attempt
        else
        {

            Document doc = getXmlDoc();

            if(usernameExists(username,doc) && credentialsMatch(
                username,oldPassword,doc))
            {
                if(newPassword.equals(newPasswordConfirm))
                {
                    changeUserPassword(username,newPassword,doc);
                    onBackPressed();
                }
                else
                {
                    userText.setText("");
                    oldPassText.setText("");
                    newPassText.setText("");
                    newPassConfirmText.setText("");

                    Toast.makeText(getApplicationContext(), "New
                        password does not match confirmation",
                        Toast.LENGTH_SHORT).show();
                }

            }
            else
            {
                if(!usernameExists(username,doc))
                    Toast.makeText(getApplicationContext(), "User
                        does not exist", Toast.LENGTH_SHORT).show()
                        ;
                else
                    Toast.makeText(getApplicationContext(), "Old
                        password does not match", Toast.
                        LENGTH_SHORT).show();

                userText.setText("");
                oldPassText.setText("");
                newPassText.setText("");
                newPassConfirmText.setText("");
            }

        }

    }

    public void handlePassChangeRequest(View view)
    {
        prev_page = current_page.LOGIN;
        setContentView(R.layout.activity_spy_app_passchange);
        cur_page = current_page.PASSCHANGE;
    }


    public void changeUserPassword(String username, String password
        , Document doc)
    {
        StringWriter writer = null;

        Element element = doc.getDocumentElement();
        boolean onUserEntry = false;

        element.normalize();

        NodeList nList = doc.getElementsByTagName("user");

        Log.i(TAG, "in password change\n\n ");

        for (int i = 0; i < nList.getLength(); i++) {

            Node node = nList.item(i);
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                if (node.getNodeName().equals("user")) {

                    Element element2 = (Element) node;
```

```java
                    NodeList userCreds = element2.getChildNodes();

                    for (int d = 0; d < userCreds.getLength(); d++) {
                        Node credList1 = userCreds.item(d);
                        //Node credList2 = userCreds.item(d+);

                        if (credList1.getNodeType() == Node.ELEMENT_NODE)
                        {
                            Element element3 = (Element) credList1;

                            if (credList1.getNodeName().equals("name"))
                            {
                                if (username.equals(element3.getTextContent()))
                                {
                                    Log.i(TAG, "name matchs!: " + element3.getTextContent());
                                    onUserEntry = true;
                                }
                            }
                            if(credList1.getNodeName().equals("pass") && onUserEntry)
                            {
                                element3.setTextContent(password);
                                onUserEntry = false;
                            }

                        }
                    }
                }
            }
        }

        // transform doc back to xml string
        writer = writeXml(doc);

        if(writeCredentialsFile(writer.toString()));
        else
            Log.i(TAG, "Error in writing encrypted file back!!");

    }
```

2.7 Bonus Functionality

Finally, we have a working spy application with a login interface, however our task had not been complete yet. We were given the option of 2 bonus functionality requirement sets. We chose the first one involving more UI related functions that will be detailed in this section.

2.7.1 Correct live camera overlay orientation

First of the bonus requirements, the camera needed to stay fixed and correctly display when rotating the tablet. To do this I offered the solution of locking the layout in the Manifest.

```
    <activity android:name=".SpyApp" android:screenOrientation="
        portrait">
```

2.7.2 Correct image view orientation

Luckily my offered solution worked for the orientation of the image view as well, since it locked the whole app to portrait mode.

2.7.3 Add hide UI button(s)to hide both live camera view and image view

This feature required us to hide everything on the camera layout with the touch of a button, I developed the code needed to hide the UI and then restore its functionality by tapping the back button with the hidden mode detected.

```


    public boolean hidden_ui = false;

    public void hide_ui(View view)
    {
        if(cur_page == current_page.CAMERA && !hidden_ui)
        {
            Button captureButton = findViewById(R.id.captureButton)
                ;
            Button recordButton = findViewById(R.id.recordButton);
            Button hide_ui_btn = findViewById(R.id.hide_ui_btn);
            FrameLayout cameraPreview = findViewById(R.id.
                camPreview);
            ImageView imgView = findViewById(R.id.imageView);
            imgView.setVisibility(View.INVISIBLE);
            cameraPreview.setVisibility(View.INVISIBLE);
            captureButton.setVisibility(View.INVISIBLE);
            recordButton.setVisibility(View.INVISIBLE);
            hide_ui_btn.setVisibility(View.INVISIBLE);
            getSupportActionBar().hide();

            hidden_ui = true;

        }
    }
    public void unhide_ui()
    {

            Button captureButton = findViewById(R.id.captureButton)
                ;
            Button recordButton = findViewById(R.id.recordButton);
            Button hide_ui_btn = findViewById(R.id.hide_ui_btn);
            FrameLayout cameraPreview = findViewById(R.id.
                camPreview);
            ImageView imgView = findViewById(R.id.imageView);
           // imgView.setVisibility(View.VISIBLE);
            cameraPreview.setVisibility(View.VISIBLE);
            captureButton.setVisibility(View.VISIBLE);
            recordButton.setVisibility(View.VISIBLE);
            hide_ui_btn.setVisibility(View.VISIBLE);
            getSupportActionBar().hide();

            hidden_ui = false;

    }

    @Override
    public void onBackPressed() {

```

```java
        if(cur_page == current_page.LOGIN)
        {
            Intent intent = new Intent(Intent.ACTION_MAIN);
            intent.addCategory(Intent.CATEGORY_HOME);
            intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
            startActivity(intent);
        }
        else if(cur_page == current_page.FAKE)
        {
            getSupportActionBar().show();

            if(prev_page == current_page.LOGIN) {
                setContentView(R.layout.activity_spy_app_login);
                cur_page = current_page.LOGIN;
                enableTextListenier();

            }
            else if(prev_page == current_page.CAMERA) {

                    drawApp();

            }
            else if(prev_page == current_page.REGISTER) {
                setContentView(R.layout.activity_spy_app_register);
                cur_page = current_page.REGISTER;
            }
            else
            {
                setContentView(R.layout.activity_spy_app_passchange
                    );
                cur_page = current_page.PASSCHANGE;
            }
        }
        else
            {
                if(cur_page == current_page.CAMERA && hidden_ui)
                {
                    unhide_ui();
                }
                else {
                    login_attempts = 3;

                    cur_page = current_page.LOGIN;
                    setContentView(R.layout.activity_spy_app_login)
                        ;

                    enableTextListenier();

                    TextView attempt_text = findViewById(R.id.
                        attempts_view);
                    attempt_text.setVisibility(View.INVISIBLE);
                    TextView attempt_num = findViewById(R.id.
                        attempt_num);
                    attempt_num.setVisibility(View.INVISIBLE);

                    Button btn = findViewById(R.id.login_confirm);
                    btn.setEnabled(true);
```

```
                        }
                    }

        }
```

2.7.4 Cover the whole UI with a screenshot from a popular app such as Facebook or Gmail

In this portion of the bonus functionality I had helped by coming up with a method to trigger a new "incognito" type layout which was provide by my group partner. If so, many taps were experienced on the touch screen at once, then this layout would be swapped to and the rest of the UI hidden as to appear as the user was playing "2048" on their device. I was able to develop the following..

```java


    @Override
    public boolean onTouchEvent(MotionEvent event) {

        if(cur_page != current_page.FAKE) {
            super.onTouchEvent(event);

            Date currentTime = Calendar.getInstance().getTime();
            Log.d(TAG, "TOUCH!! cur time:" + currentTime + "\n\n");
            if (currentTime == currentTime)
                touchCount++;
            if (touchCount > 25) {
                prev_page = cur_page;
                setContentView(R.layout.activity_spy_app_fake);
                cur_page = current_page.FAKE;


                ImageView image = findViewById(R.id.imageView3);

                image.setImageResource(R.drawable.screen);
                getWindow().setFlags(WindowManager.LayoutParams.
                    FLAG_FULLSCREEN,
                        WindowManager.LayoutParams.FLAG_FULLSCREEN)
                            ;
                getSupportActionBar().hide();

                image.getLayoutParams().height = ViewGroup.
                    LayoutParams.MATCH_PARENT;
                image.getLayoutParams().width = ViewGroup.
                    LayoutParams.MATCH_PARENT;
                image.setAdjustViewBounds(false);
                image.setScaleType(ImageView.ScaleType.FIT_XY);

                touchCount = 0;
            }
            lasttime = currentTime;
        }
        return false;
    }
```

2.7.5 Make ‘Register’ button disabled after user inputs their username and password

In making the registration button on the login screen unusable, my role was to develop code to detect when the user input their credentials. However i needed to also disable the registration button when both username and password were filled at the same time. To do this I crafted an on text change listener that always checked both fields.

```


    public void enableTextListenier()
    {

        Log.d(TAG,"in listener!");

        final EditText username_et = findViewById(R.id.
            login_username);
        final EditText password_et = findViewById(R.id.
            login_password);
        final Button regButton = findViewById(R.id.register_enter);

        //final boolean reg_btn_allow = false;

        username_et.addTextChangedListener(new TextWatcher() {

            @Override
            public void onTextChanged(CharSequence s, int start,
                int before,
                                      int count) {
                // TODO Auto-generated method stub

                if (s.toString().equals("") && password_et.getText
                    ().toString().equals("")) {
                    Log.d(TAG,"TEXT IS EMPTY!");
                   // reg_btn_allow = true;
                    regButton.setVisibility(View.VISIBLE);
                } else {
                    Log.d(TAG,"TEXT IS something!");
                    //reg_btn_allow = false;
                    regButton.setVisibility(View.INVISIBLE);
                }
            }

            @Override
            public void beforeTextChanged(CharSequence s, int start
                , int count,
                                          int after) {
                // TODO Auto-generated method stub

            }

            @Override
            public void afterTextChanged(Editable s) {
                // TODO Auto-generated method stub

            }
        });

        password_et.addTextChangedListener(new TextWatcher() {

            @Override
            public void onTextChanged(CharSequence s, int start,
                int before,
                                          int count) {
                // TODO Auto-generated method stub

                if (s.toString().equals("") && username_et.getText
                    ().toString().equals("")) {
                    Log.d(TAG,"TEXT IS EMPTY!");
                    // reg_btn_allow = true;
                    regButton.setVisibility(View.VISIBLE);
                } else {
                    Log.d(TAG,"TEXT IS something!");
                    //reg_btn_allow = false;
                    regButton.setVisibility(View.INVISIBLE);
                }
            }

            @Override
            public void beforeTextChanged(CharSequence s, int start
                , int count,
                                              int after) {
                // TODO Auto-generated method stub

            }

            @Override
            public void afterTextChanged(Editable s) {
                // TODO Auto-generated method stub

            }
        });

    }
```

2.7.6 Changing the user password requires verifying current password

Much of the backend work for this requirement I had already personally developed, so it didn't take much more work to verify the user's credentials in order to grant them access to change their password. In the code below I did this along with other field verification for current username, current password, new password, and confirm new password.

```


    public void handlePassChangeAttempt(View view)
    {
        EditText userText = (EditText)findViewById(R.id.
            passchg_username);
        EditText oldPassText = (EditText)findViewById(R.id.
            passchg_prevpass);
        EditText newPassText = (EditText)findViewById(R.id.
            passchg_newpass);
        EditText newPassConfirmText = (EditText)findViewById(R.id.
            passchg_newpass_confirm);
        String username = userText.getText().toString();
        String oldPassword = oldPassText.getText().toString();
        String newPassword = newPassText.getText().toString();
        String newPasswordConfirm = newPassConfirmText.getText().
            toString();

        // gain access to filesystem write priv first
        if(checkAppPermissions()) {
            requestAppPermissions();

            userText.setText("");
            oldPassText.setText("");
            newPassText.setText("");
            newPassConfirmText.setText("");
        }
            // handle pass change attempt
        else
        {

            Document doc = getXmlDoc();

            if(usernameExists(username,doc) && credentialsMatch(
                username,oldPassword,doc))
            {
                if(newPassword.equals(newPasswordConfirm))
                {
                    changeUserPassword(username,newPassword,doc);
                    onBackPressed();
                }
                else
                {
                    userText.setText("");
                    oldPassText.setText("");
                    newPassText.setText("");
                    newPassConfirmText.setText("");

```

```java
                    Toast.makeText(getApplicationContext(), "New password does not match confirmation", Toast.LENGTH_SHORT).show();
                }

            }
            else
            {
                if(!usernameExists(username,doc))
                    Toast.makeText(getApplicationContext(), "User does not exist", Toast.LENGTH_SHORT).show();
                else
                    Toast.makeText(getApplicationContext(), "Old password does not match", Toast.LENGTH_SHORT).show();

                userText.setText("");
                oldPassText.setText("");
                newPassText.setText("");
                newPassConfirmText.setText("");
            }

        }

    }
```

3 Android OS Customization

After we had completed our work on developing a "Spy App" for and android device. We were then tasked with compiling/installing/and modifying Lineage OS code for our tablet. The compilation and setup took place on my machine, and the modification of the boot animation was performed with the help of my partner.

3.1 Android Source Code Download Setup

To start, I needed to download the source for Lineage OS on my Linux 16.04 virtual machine. Using `repo sync` I was able to do this but ended up missing proprietary blobs when i went to compile. So to fix this, I used a repo on GitHub known as TheMuppets located here.

`https://github.com/TheMuppets/proprietary_vendor_asus/tree/lineage-16.0/flo`

After acquiring the repo I was able to re sync the source code with it after editing the file `.repo/local_manifests/roomservice.xml` with the correct blobs repo.

```
<project name="TheMuppets/proprietary_vendor_asus" path="vendor/
    asus" depth="1" />
```

Also running `breakfast flo` again resolved any errors about missing blob files.

3.2 Compile Android Source Code

Now that all the necessary code has been collected for compiling, I was able to compile without any errors regarding missing sources. However new errors were experienced regarding my virtual machines' memory parameters, as well as environmental variables for the jack server. The jack server is an android toolchain needed to compile the Java source code for our target OS and in my case it didn't have enough memory to continue. By allocated it more memory via setting the correct env variables I was able to get the image to finally compile. This just required me updating my bashrc file for my current user.

```
export ANDROID_JACK_VM_ARGS="-Dfile.encoding=UTF-8 -XX:+
    TieredCompilation -Xmx4G"
export JACK_SERVER_VM_ARGUMENTS="-Dfile.encoding=UTF-8 -XX:+
    TieredCompilation -Xmx4g"
USE_CCACHE=1
ccache -M 50G
```

Also i would need to restart the jack server to apply the changes.

```
jack-admin kill-server
jack-admin start-server
```

3.3 Custom Boot Animation

With the final Lineage OS image in hand, I could now install it on the tablet to test it. To complete the task however, we had to change the boot animation on the device. With help of my partner, he supplied me with a customized boot animation he had modified to get sound and I was able to replace within the lineage OS directory..

```
android/lineage/out/target/product/flo/system/media/bootanimation.zip
```

Afterwards I rebuilt the image and installed it through the recovery menu as well as tested. Everything worked fine and our task had been completed.