CHAPTER 7
SOFTWARE TESTING

7.1 SYSTEM TEST

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results.

7.1.1 White Box Testing

- Execution of every path in the program.

7.1.2 Black Box Testing

- Exhaustive input testing is required to find all errors

.

7.2 TYPES OF TESTS

- White Box Testing
- Black Box Testing
- Unit Testing
- Functional Testing
- Performance Testing
- Integration Testing
- Objective
- Integration Testing
- Validation Testing
- System Testing
- Structure Testing
- Output Testing
- User Acceptance Testing

7.2.1 Unit Testing

- Unit testing, also known as Module Testing, focuses verification efforts on the module. The module is tested separately and this is carried out at the programming stage itself.
- Unit Test comprises of the set of tests performed by an individual programmer before integration of the unit into the system.
- Unit test focuses on the smallest unit of software design- the software component or module.
- Using component level design, important control paths are tested to uncover errors within the boundary of the module.
- Unit test is white box oriented and the step can be conducted in parallel for multiple components.

7.2.2 Functional Testing:

- Functional test cases involve exercising the code with normal input values for which the expected results are known, as well as the boundary values

Objective:

- The objective is to take unit-tested modules and build a program structure that has been dictated by design.

7.2.3 Performance Testing:

- Performance testing determines the amount of execution time spent in various parts of the unit, program throughput, and response time and device utilization of the program unit. It occurs throughout all steps in the testing process.

7.2.4 Integration Testing:

- It is a systematic technique for constructing the program structure while at the same time conducting tests to uncover errors associated with in the interface.
- It takes the unit tested modules and builds a program structure.
- All the modules are combined and tested as a whole.
- Integration of all the components to form the entire system and a overall testing is executed.

7.2.5 Validation Testing:

- Validation test succeeds when the software functions in a manner that can be reasonably expected by the client.
- Software validation is achieved through a series of black box testing which confirms to the requirements.
- Black box testing is conducted at the software interface.
- The test is designed to uncover interface errors, is also used to demonstrate that software functions are operational, input is properly accepted, output are produced and that the integrity of external information is maintained.

7.2.6 System Testing:

- Tests to find the discrepancies between the system and its original objective, current specifications and system documentation.

7.2.7 Structure Testing:

- It is concerned with exercising the internal logic of a program and traversing particular execution paths.

7.2.8 Output Testing:

- Output of test cases compared with the expected results created during design of test cases.
- Asking the user about the format required by them tests the output generated or displayed by the system under consideration.
- Here, the output format is considered into two was, one is on screen and another one is printed format.
- The output on the screen is found to be correct as the format was designed in the system design phase according to user needs.
- The output comes out as the specified requirements as the user's hard copy.

7.2.9 User acceptance Testing:

- Final Stage, before handling over to the customer which is usually carried out by the customer where the test cases are executed with actual data.
- The system under consideration is tested for user acceptance and constantly keeping touch with the prospective system user at the time of developing and making changes whenever required.
- It involves planning and execution of various types of test in order to demonstrate that the implemented software system satisfies the requirements stated in the requirement document.

CHAPTER 8
CONCLUSION

For mobile multimedia streaming services, how to provide appropriate multimedia files according to the network and hardware devices is an interesting subject. In this study, a set of adaptive networks and a device aware QoS approach for interactive mobile streaming was proposed. The DNEM and DBPM were used for the prediction of network and hardware features, and the communication frequency and SVC multimedia streaming files most suitable for the device environment were determined according to these two modules. In the experiment, the overall prototype architecture was realized and an experimental analysis was carried out. The experimental data proved that the method could maintain a certain level of multimedia service quality for dynamic network environments and ensure smooth and complete multimedia streaming services. Cloud services may accelerate research on SVC coding in the future. this study presented a network and device-aware Quality of Service (QoS) approach that provides multimedia data suitable for a terminal unit environment via interactive mobile streaming services, further considering the overall network environment and adjusting the interactive transmission frequency and the dynamic multimedia transcoding, to avoid the waste of bandwidth and terminal power. Finally, this study realized a prototype of this architecture to validate the feasibility of the proposed method.

CHAPTER 9
FUTURE ENHANCEMENT

In this work, we just consider a single flow scenario and ignore the interference from the other flows as well as the competitive bidding for spectrum usage from the other flows. In a CRN with multi flows, the CR source nodes need to develop sophisticated bidding strategies considering the competition from the peer flows, and the SSP should jointly consider the cross-layer factors and the bidding values to determine the sharing of the harvested spectrum.

}

APPENDICES

APPENDIX 1-SAMPLE CODING

//NEW WEB SERVICE

```
@WebService(serviceName = "NewWebService")
public class NewWebService {


   /**
    * Web service operation
    */
@WebMethod(operationName = "signin")
public String signin(@WebParam(name = "username") String username,
@WebParam(name = "password") String password) {
try {
Common_DB cd=new Common_DB();
ResultSetrs=Common_DB.LoginCheck("psjav05",
"login","username","password", username, password);
if(rs.next()) {
return "success";
            }
else {
return "username or password is invalid"; }
        } catch (Exception ex) {
Logger.getLogger(NewWebService.class.getName()).log(Level.SEVERE, null,
ex);
return "server temporarily not available";   }
}
```

```
/**Web service operation
@WebMethod(operationName = "signup")
public String signup(@WebParam(name = "username") String username,
@WebParam(name = "password") String password, @WebParam(name =
"email") String email) { try {
Common_DB cd=new Common_DB();
intrs=Common_DB.InsertTable("psjav05", "INSERT INTO
login(username,password,email)
VALUES('"+username+"','"+password+"','"+email+"')");
if(rs>0) {
return "success";  }
else {
return "username is already available"; }
 } catch (Exception ex) {
Logger.getLogger(NewWebService.class.getName()).log(Level.SEVERE, null,
ex);
return "server temporarily not available"; }   }
```

//TRANSCODER

```
public static void main(String a[]) { try {
 File source=new File("E:/Wildlife.wmv");
 File target=new File("E:/my.mp4");
AudioAttributesaattrib=new AudioAttributes();
aattrib.setCodec("libfaac");
aattrib.setBitRate(new Integer(256000));
aattrib.setSamplingRate(new Integer(65100));
aattrib.setChannels(new Integer(2));
VideoAttributesvattrib=new VideoAttributes();
vattrib.setCodec("mpeg4");
vattrib.setBitRate(new Integer(3200000));
vattrib.setFrameRate(new Integer(18));
vattrib.setSize(new VideoSize(620, 480));
EncodingAttributesattrs = new EncodingAttributes();
attrs.setFormat("mp4");
attrs.setAudioAttributes(aattrib);
attrs.setVideoAttributes(vattrib);
                    Encoder encoder = new Encoder();
encoder.encode(source, target, attrs);
        } catch (IllegalArgumentException ex) {
Logger.getLogger(TransCoder.class.getName()).log(Level.SEVERE, null, ex);
        } catch (InputFormatException ex) {
Logger.getLogger(TransCoder.class.getName()).log(Level.SEVERE, null, ex);
        } catch (EncoderException ex) {
Logger.getLogger(TransCoder.class.getName()).log(Level.SEVERE, null, ex);
        }
    }
}
```

/CALL SERVICES

```
public static String getVideosFromMobileInfo(String
bandwidth,Stringphonetype,Stringnetworkname,Stringsimstate,Stringnetworkty
pe,Stringosversion,Stringusername,String method) {
		String list = null;
		SoapObject soap=new SoapObject(NAMESPACE,method);
		soap.addProperty("bandwidth",bandwidth);
		soap.addProperty("networkname",networkname);
		soap.addProperty("phonetype",phonetype);
		soap.addProperty("simstate",simstate);
		soap.addProperty("networktype",networktype);
		soap.addProperty("osversion",osversion);
		soap.addProperty("username",username);
		SoapSerializationEnvelope envelope=new
SoapSerializationEnvelope(SoapEnvelope.VER11);
		envelope.setOutputSoapObject(soap);
		HttpTransportSE http=new HttpTransportSE(URL);
		try {
			http.call(NAMESPACE+method, envelope);
			SoapPrimitivepri=(SoapPrimitive) envelope.getResponse();
			list=pri.toString();
		}
```

```
                    catch (IOException e) {
                      e.printStackTrace();
                      return list;
               } catch (XmlPullParserException e) {
                      e.printStackTrace();
                      return list;
               } return list; }
```

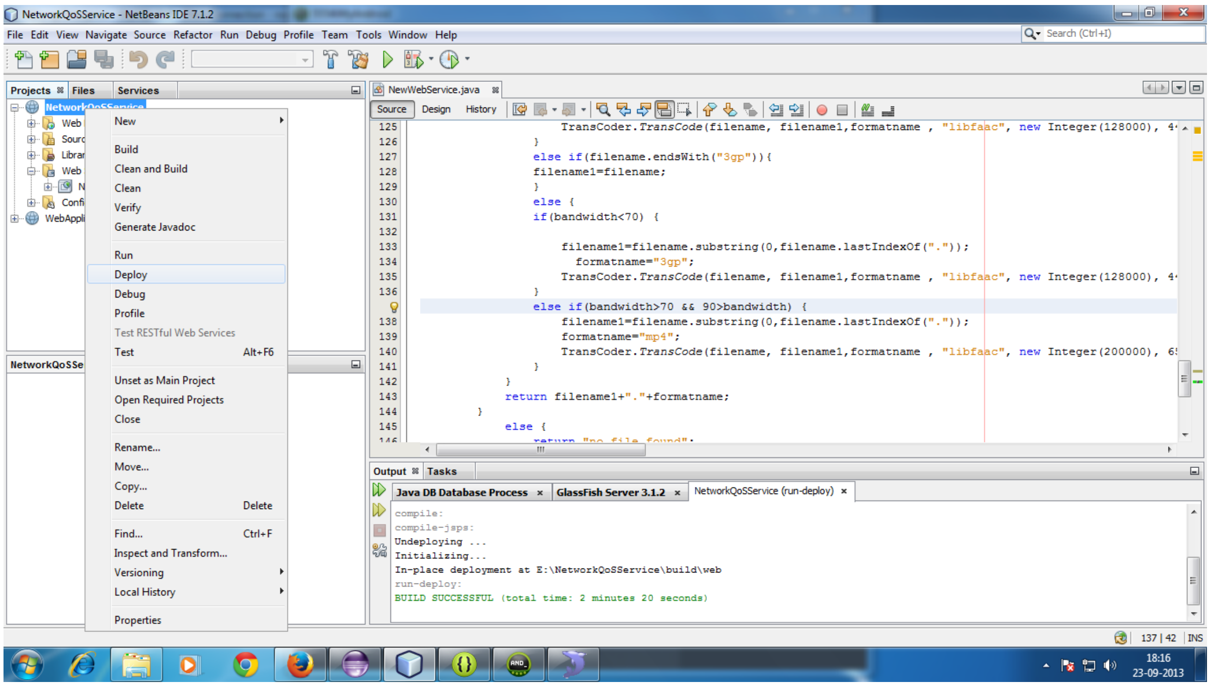
//MAIN ACTIVITY

```
protected void onCreate(Bundle savedInstanceState) {
               super.onCreate(savedInstanceState);
               setContentView(R.layout.activity_main);
               pb=(ProgressBar)findViewById(R.id.progressBar2);
               bn=(Button)findViewById(R.id.bt);
               bw=(TextView)findViewById(R.id.bandwidth);
               pt=(TextView)findViewById(R.id.phone_type);
               nt=(TextView)findViewById(R.id.network_type);
               ss=(TextView)findViewById(R.id.sim_state);
               nn=(TextView)findViewById(R.id.network_name);
               uname=(TextView)findViewById(R.id.uname);
               username=getIntent().getStringExtra("username");
               uname.setText("welcome "+username+",");
               icon=R.drawable.ic_launcher;
               bn.setOnClickListener(new OnClickListener() {
@Override
public void onClick(View v) {
```

```
// Get the telephony system service to find out network details
                        finalTelephonyManager tm = (TelephonyManager)
                        getSystemService(TELEPHONY_SERVICE);

                  // Update text views with readable values.
                  updateViews(tm);

                   // Since these attributes can change, we will register a
                // {@code PhoneStateListener} to listen for these changes and
               // update the view.
                  tm.listen(new PhoneStateListener() {
                             @Override
                  public void onServiceStateChanged(ServiceStateserviceState) {
                                 // Update our TextViews
                  updateViews(tm);
                             }

                             @Override
                  public void onDataConnectionStateChanged(int state) {
                                 // A change in data connection state may be due to
                                 // of a different network type
                  updateViews(tm);
                             }

                         },
 PhoneStateListener.LISTEN_SERVICE_STATE |
PhoneStateListener.LISTEN_DATA_CONNECTION_STATE);

                        // calculate bandwidth
                                       Downloadfiledf=new Downloadfile();
                                                 df.execute  } ); }
//         @Override
//            protected void onResume() {
//        super.onResume();
//                final TelephonyManager tm = (TelephonyManager)
getSystemService(TELEPHONY_SERVICE);
//
//                // Update text views with readable values.
//        updateViews(tm);
//            }
@Override
          publicbooleanonCreateOptionsMenu(Menu menu) {
          // Inflate the menu; this adds items to the action bar if it is present.
                    getMenuInflater().inflate(R.menu.main, menu);
                    return true;
          }
           /**
     * Update text views with telephony attributes.
     */private final void updateViews(TelephonyManager tm) {
```

```
 // The telephony system service returns integer constants for various
        // telephony attributes.
simstate=tm.getSimSerialNumber();
phonetype=mapDeviceTypeToName(tm.getPhoneType());
networkname=tm.getNetworkOperatorName();
networktype= mapNetworkTypeToName(tm.getNetworkType());
osversion=Build.VERSION.RELEASE;
ss.setText("SIM State: " + simstate);
nt.setText("Network Type: " +networktype);
pt.setText("Phone Type: " + phonetype);
nn.setText("Network Operator: " +networkname);
  }
```

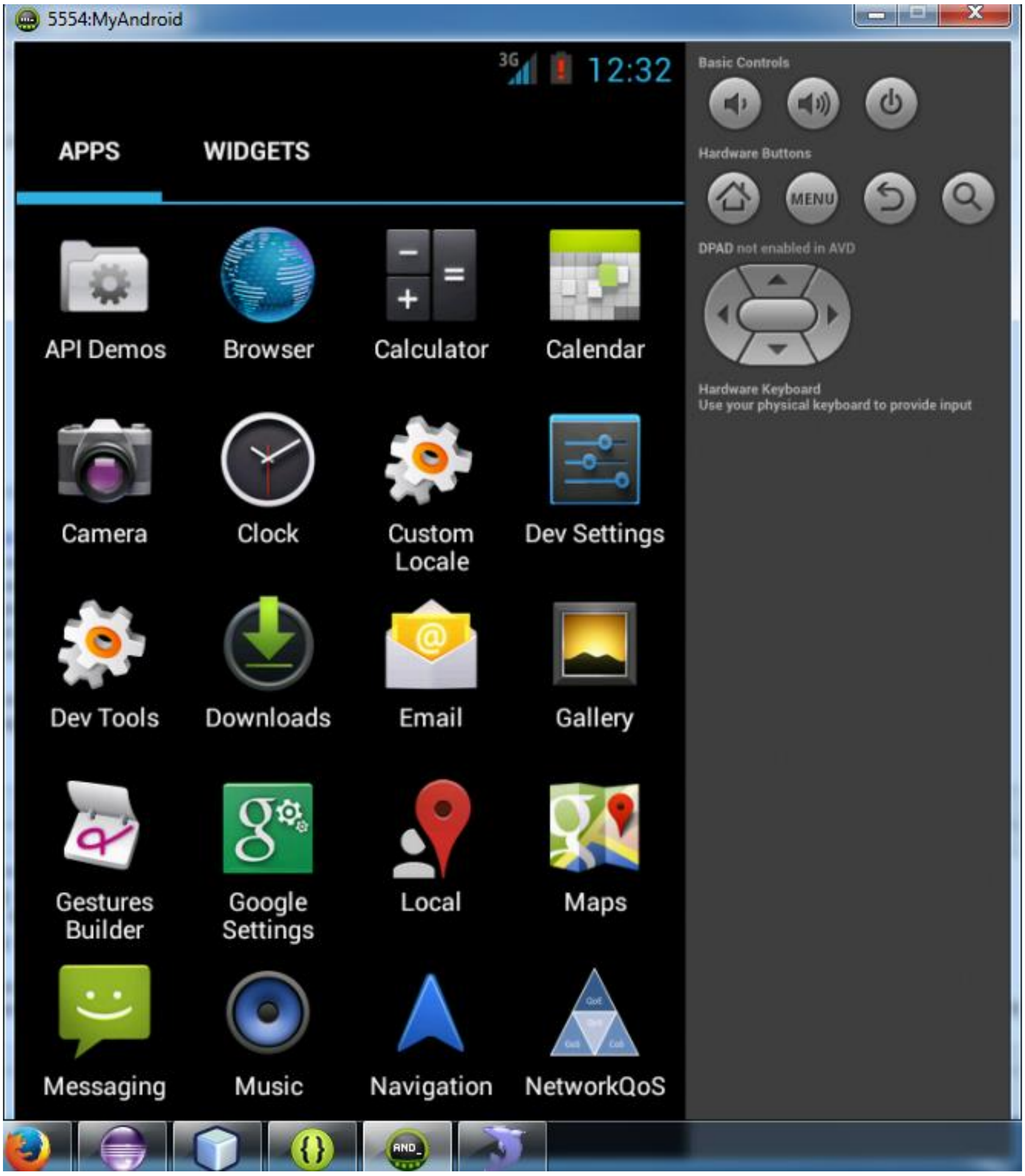
APPENDIX 2-SCREEN SHOTS



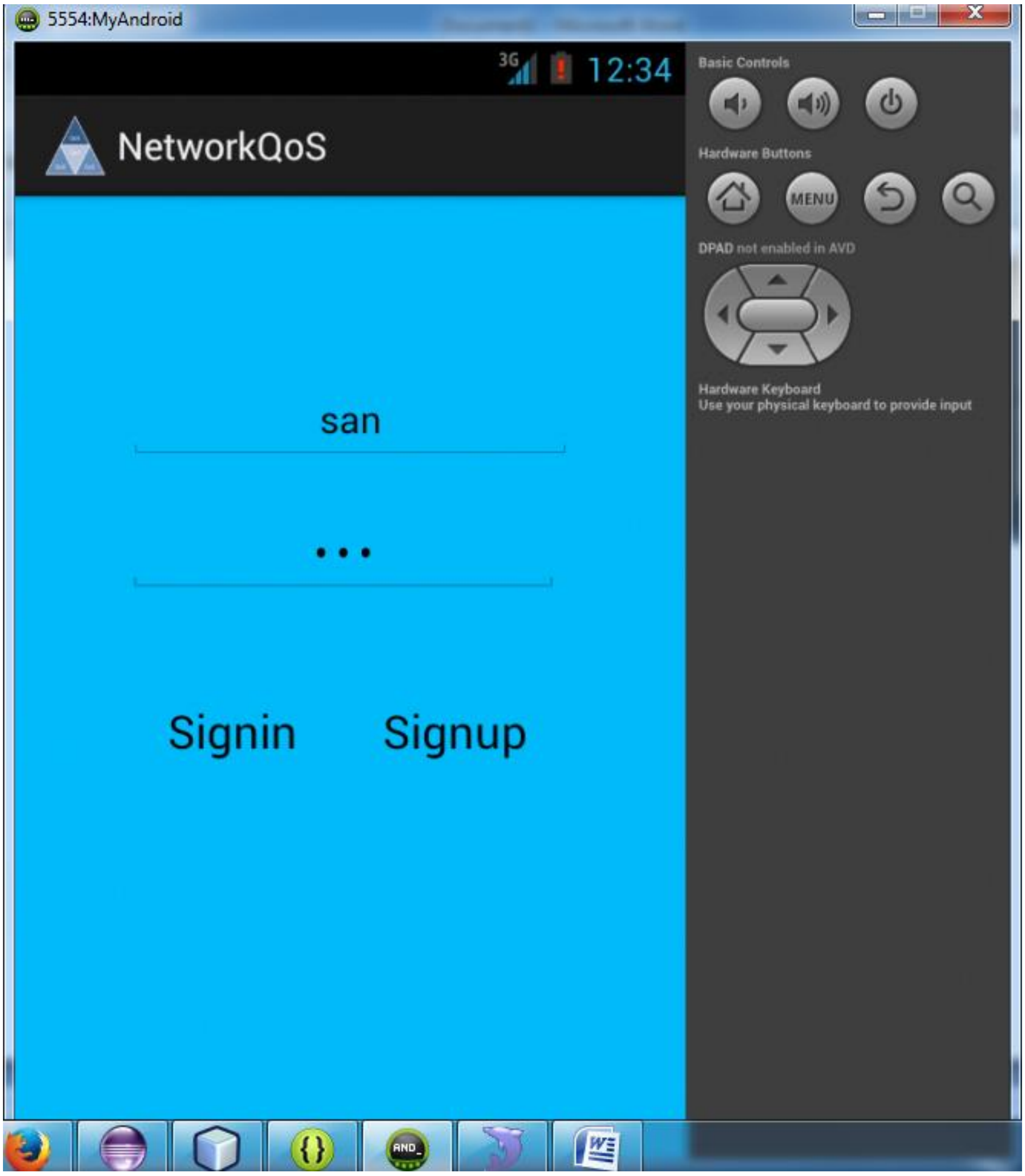
```java
                TransCoder.TransCode(filename, filename1,formatname , "libfaac", new Integer(128000),
            }
            else if(filename.endsWith("3gp")){
            filename1=filename;
            }
            else {
            if(bandwidth<70) {

                filename1=filename.substring(0,filename.lastIndexOf("."));
                  formatname="3gp";
                TransCoder.TransCode(filename, filename1,formatname , "libfaac", new Integer(128000),
            }
            else if(bandwidth>70 && 90>bandwidth) {
                filename1=filename.substring(0,filename.lastIndexOf("."));
                formatname="mp4";
                TransCoder.TransCode(filename, filename1,formatname , "libfaac", new Integer(200000),
            }
        }
        return filename1+"."+formatname;
    }
        else {
            return "no file found";
```

```
compile:
compile-jsps:
Undeploying ...
Initializing...
In-place deployment at E:\NetworkQoSService\build\web
run-deploy:
BUILD SUCCESSFUL (total time: 2 minutes 20 seconds)
```

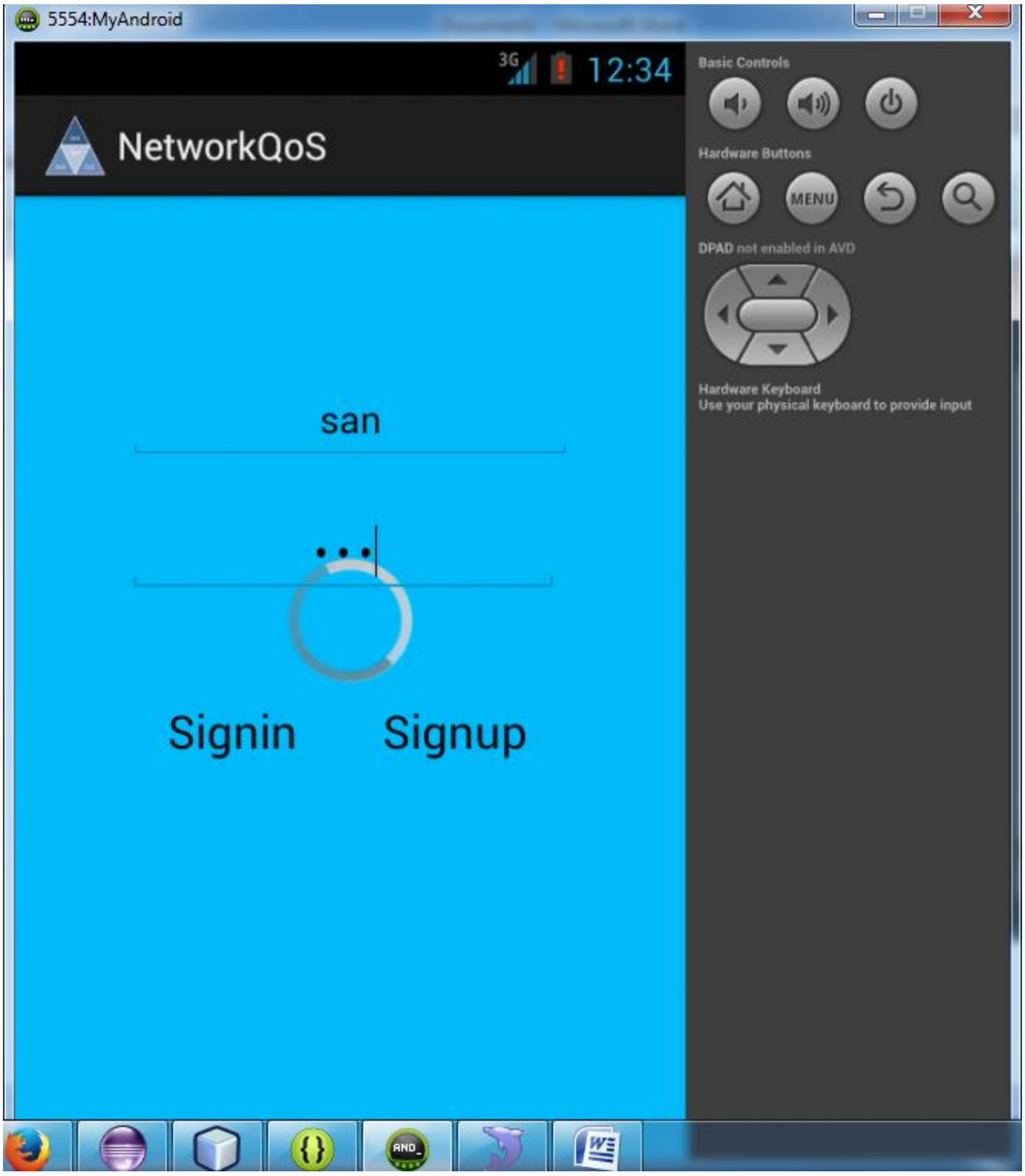
BUILDING PROCESS


5554:MyAndroid
12:32
APPS
WIDGETS
API Demos
Browser
Calculator
Calendar
Camera
Clock
Custom Locale
Dev Settings
Dev Tools
Downloads
Email
Gallery
Gestures Builder
Google Settings
Local
Maps
Messaging
Music
Navigation
NetworkQoS
Basic Controls
Hardware Buttons
MENU
DPAD not enabled in AVD
Hardware Keyboard
Use your physical keyboard to provide input

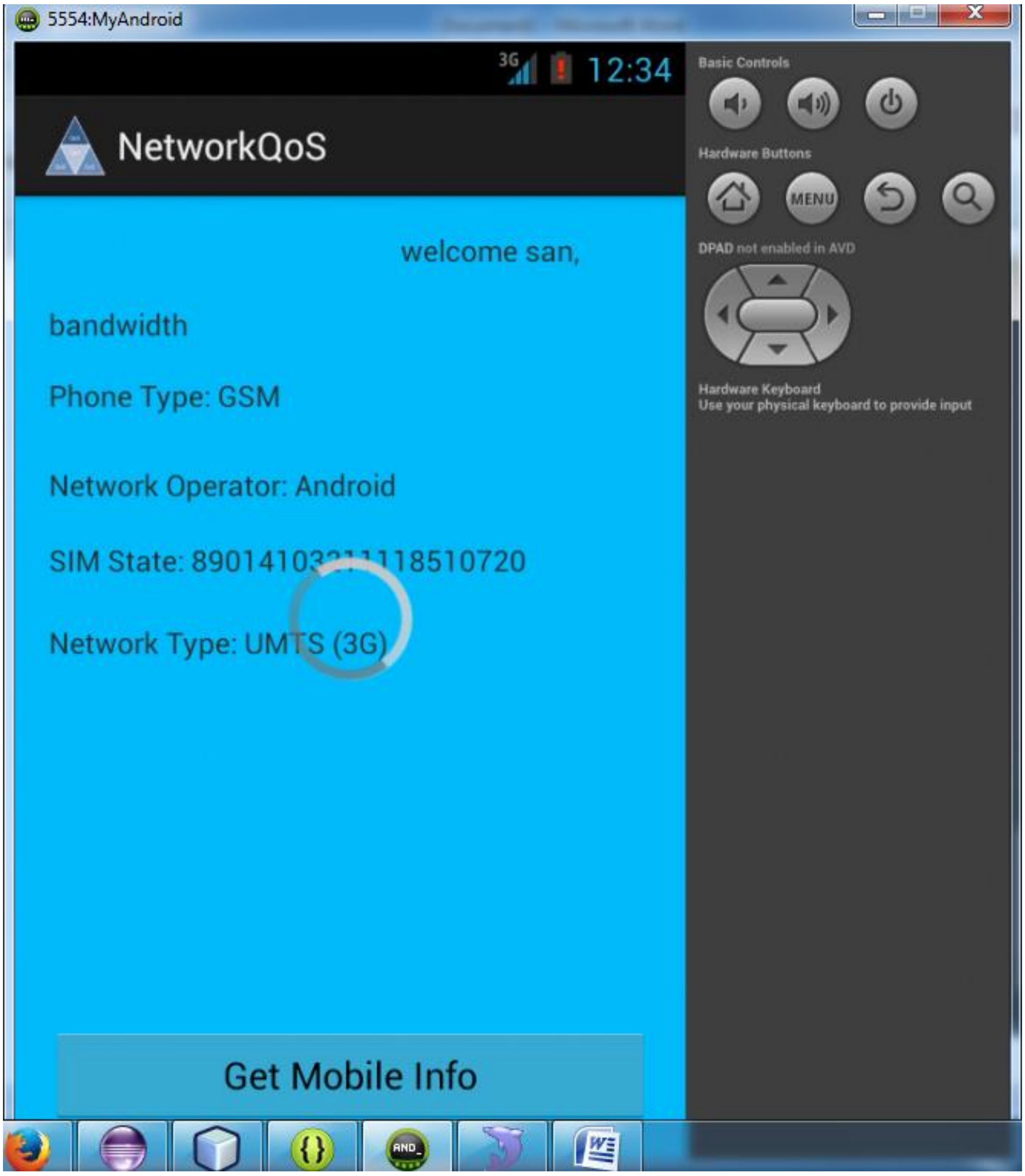
MENU SCREEN



LOGIN PROCESS



PROCESSING INFORMATION



5554:MyAndroid

NetworkQoS

welcome san,

bandwidth

Phone Type: GSM

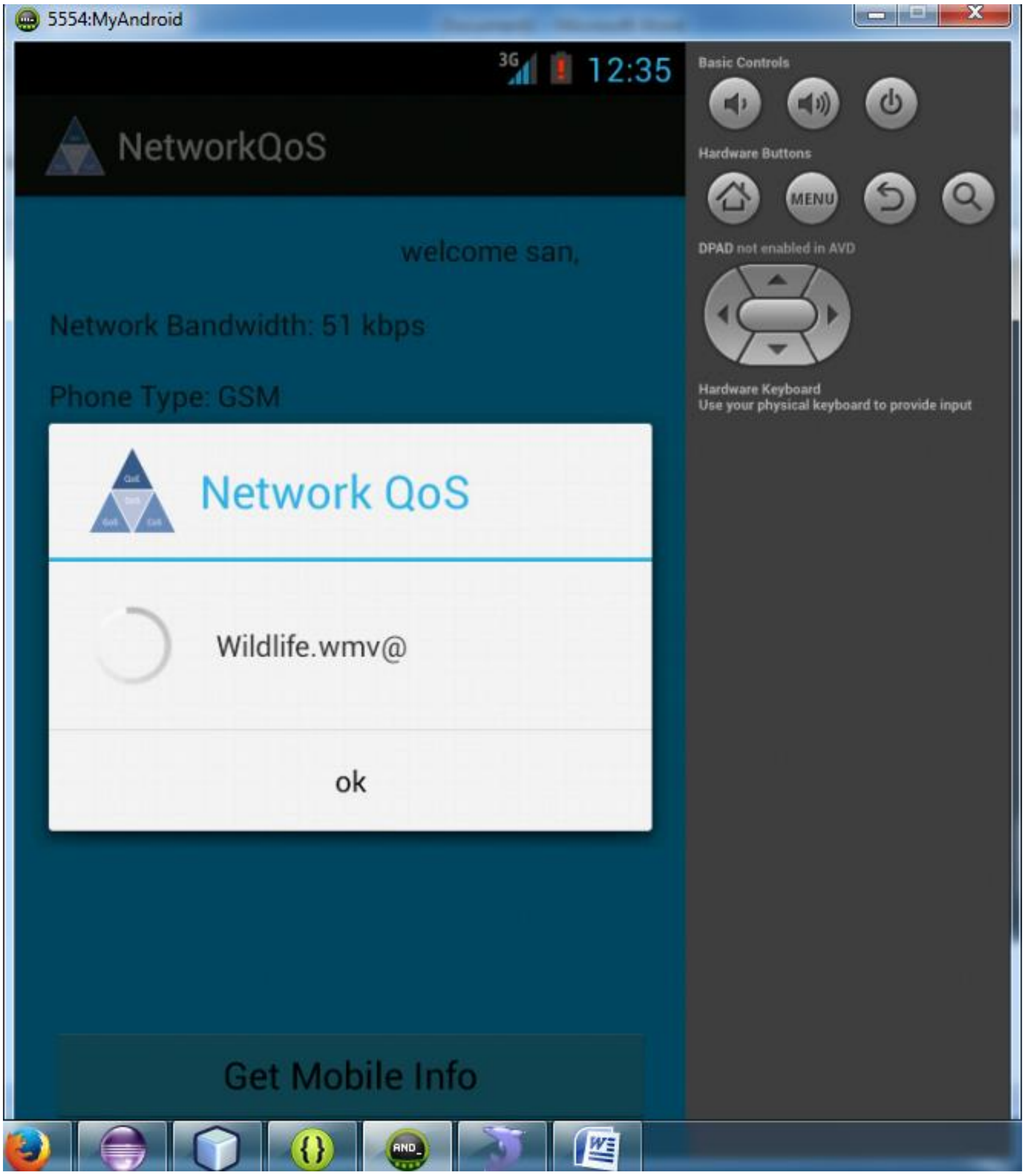
Network Operator: Android

SIM State: 89014103211118510720

Network Type: UMTS (3G)

Get Mobile Info

MOBILE INFORMATION



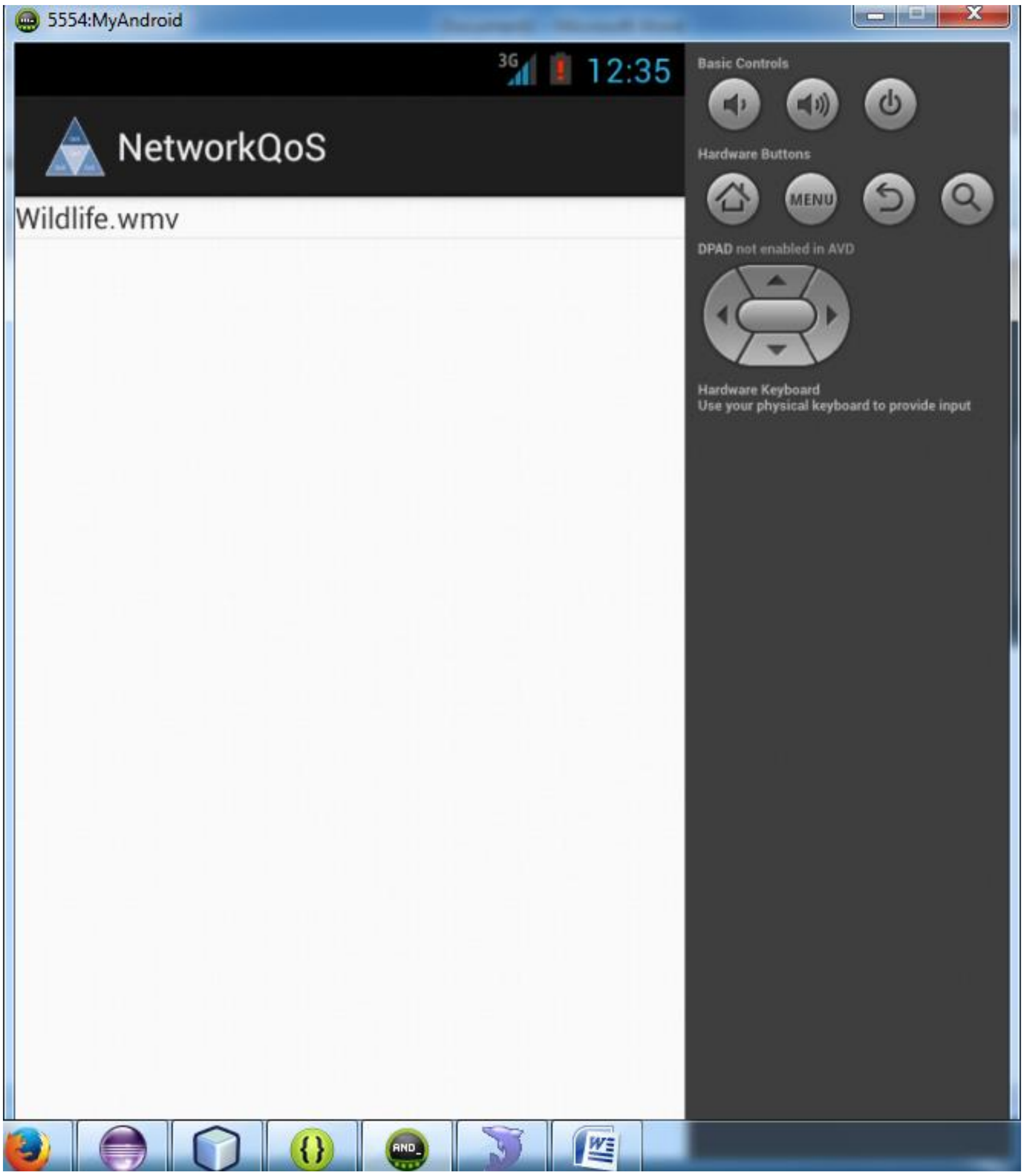
welcome san,

Network Bandwidth: 51 kbps

Phone Type: GSM

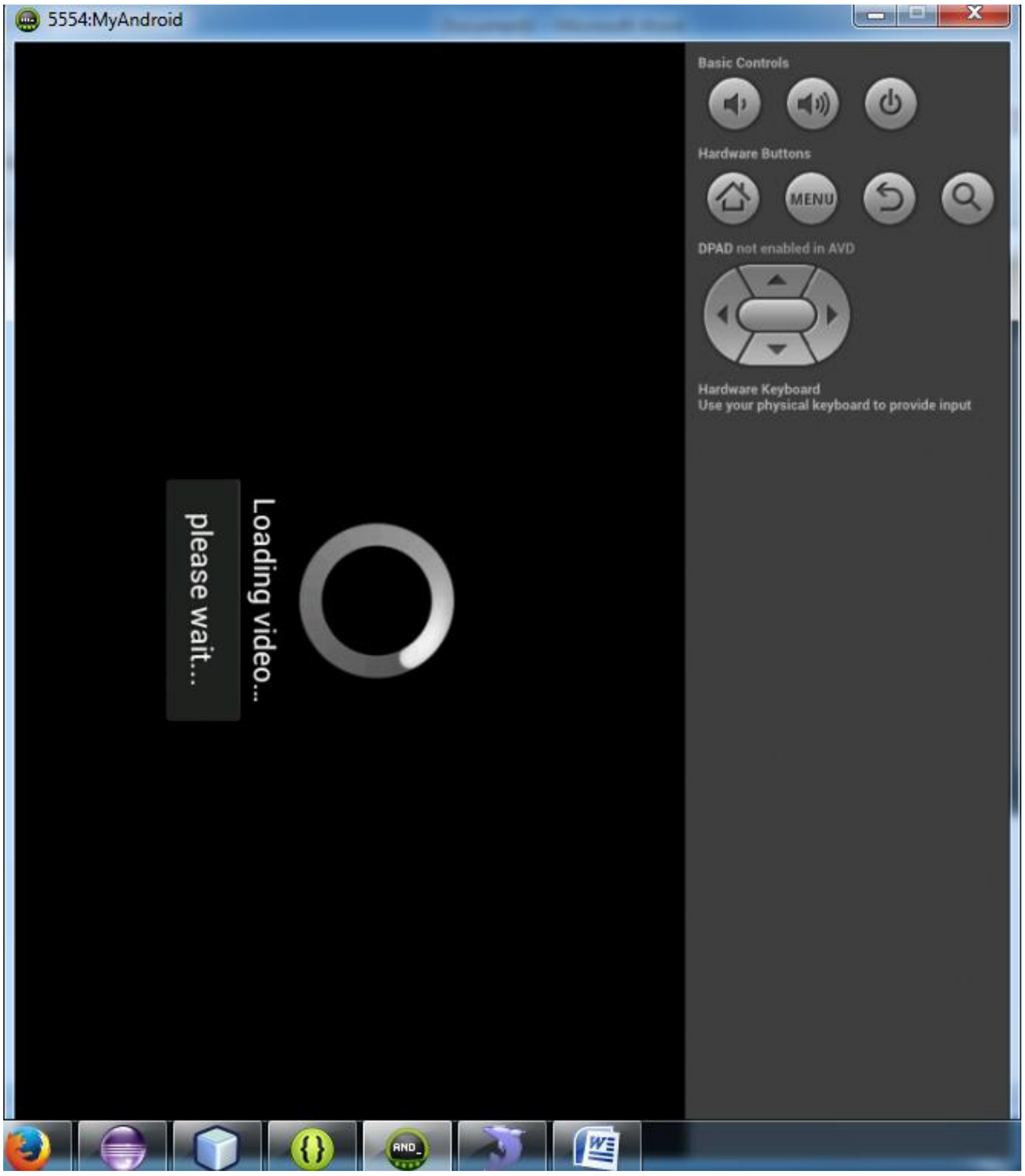
Wildlife.wmv@

UPLOADING VIDEO



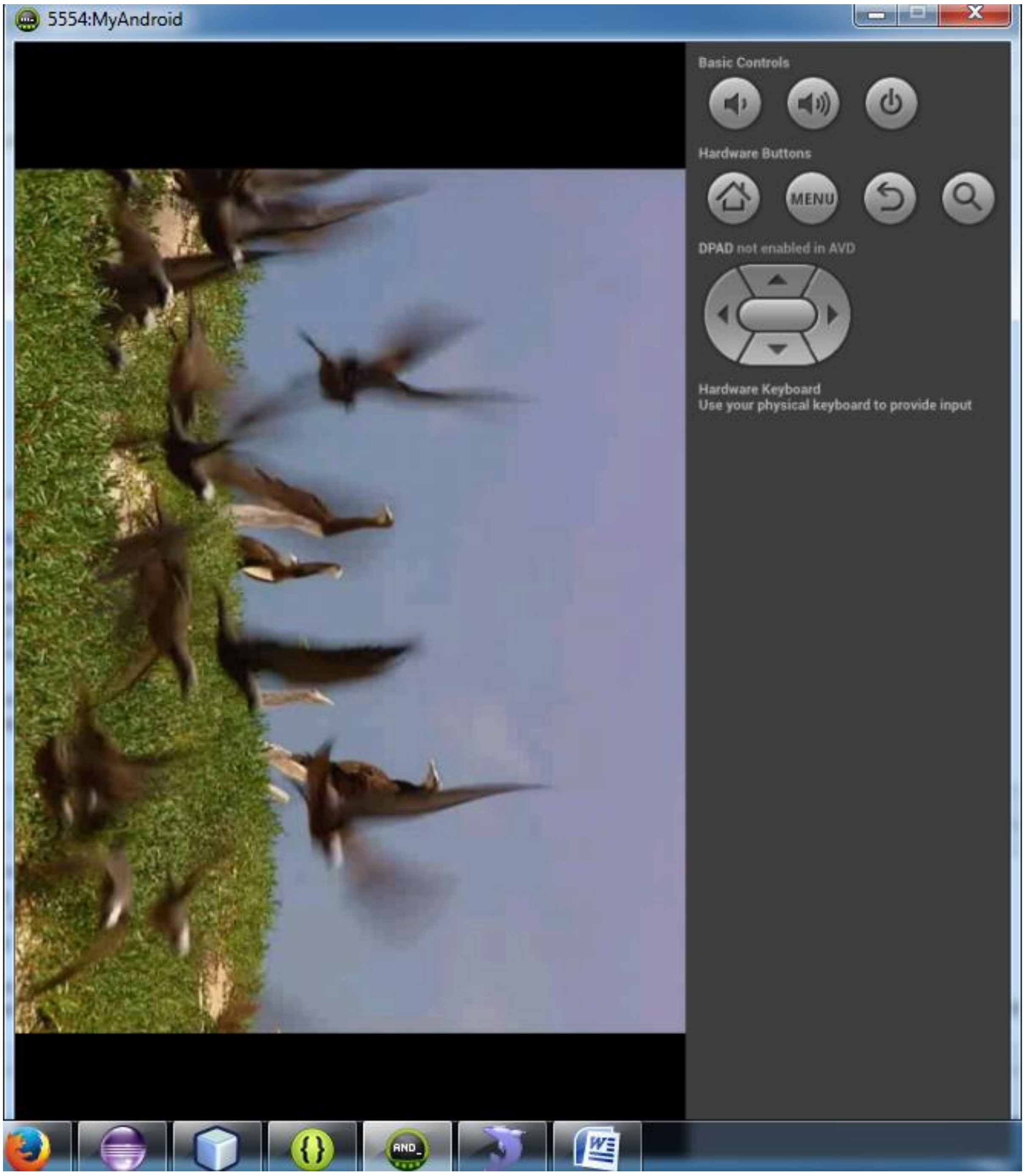
Wildlife.wmv

UPLOADED VIDEO



Loading video...
please wait...

LOADING THE VIDEO



VIDEO WITHOUT BUFFER

REFERENCES

[1] F. Tan and X. Su, “Media cloud:When media revolution meets rise of cloud computing,” in *Proc. IEEE 6th Int. Symp. Service Oriented Syst. Eng., 2011, pp. 251–261.*

[2] G. Q. Hu, W. P. Tay, and Y. G. Wen, “Cloud robotics: Architecture, challenges and applications,” *IEEE Network, Special Issue on Machine and Robotic Networking, vol. 26, no. 3, pp. 21–28, May-Jun. 2012.*

[3] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph,R.Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, “A view of cloud computing,” *Commun. ACM, vol. 53, p. 508, Apr. 2010.*

[4] S. Ferretti, V. Ghini, F. Panzieri, and E. Turrini, “Seamless support of multimedia distributed applications through a cloud,” in *Proc. IEEE 3rd Int. Conf. Cloud Comput. (CLOUD), 2010, pp. 548–549.*

[5] W. Zhu, C. Luo, J. F. Wang, and S. P. Li, “Multimedia cloud computing,” *IEEE Signal Process. Mag., vol. 28, no. 3, pp. 59–69, 2011.*