

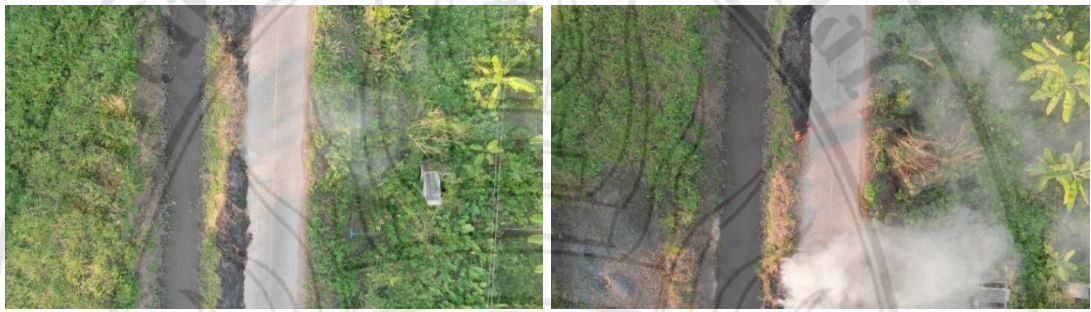
บทที่ 3

วิธีดำเนินการวิจัย

โครงการวิจัยเรื่อง โดรนบินอัตโนมัติตรวจสอบการเกิดไฟฟ้า กรณีศึกษาเทศบาลเมืองเมือง
แกนพัฒนา จังหวัดเชียงใหม่ มีวัตถุประสงค์เพื่อสร้างต้นแบบโดรนบินอัตโนมัติตรวจสอบการเกิดไฟฟ้า
ในเขตชุมชนเทศบาลเมืองเมืองแกนพัฒนา อำเภอแม่แตง จังหวัดเชียงใหม่ เพื่อสร้างระบบแจ้งเตือน
พิกัดไฟฟ้าของชุมชนเทศบาลเมืองเมืองแกนพัฒนา อำเภอแม่แตง จังหวัดเชียงใหม่ เพื่อประเมินความ
พึงพอใจของการใช้งานระบบของชุมชนเทศบาลเมืองเมืองแกนพัฒนา อำเภอแม่แตง จังหวัดเชียงใหม่
ในส่วนของบทที่ 3 นี้จะเป็นเนื้อหาที่เกี่ยวข้องกับขั้นตอนการดำเนินงานวิจัย ประกอบด้วย ขั้นตอน
เก็บรวบรวมข้อมูลที่ใช้ในการดำเนินการวิจัย ขั้นตอนการวิเคราะห์และออกแบบระบบการตรวจสอบ
ไฟฟ้า วิธีการสร้างโมเดลตรวจสอบไฟฟ้า การออกแบบหน้าจอบนแอปพลิเคชันควบคุมโดรนเพื่อ
ตรวจสอบไฟฟ้า วิธีการนำโมเดลที่ได้จากการเทรนให้รู้จักภาพไฟมาใส่แอนดรอยด์แอปพลิเคชัน และ
วิธีการเก็บภาพไฟป่าลงในฐานข้อมูล การสร้างระบบแจ้งเตือน โดยมีรายละเอียดของการดำเนินการ
วิจัยดังต่อไปนี้

3.1 ขั้นตอนเก็บรวบรวมข้อมูลที่ใช้ในการดำเนินการวิจัย

ขั้นตอนการเก็บรวบรวมข้อมูลที่ใช้ในการตรวจสอบภาพไฟฟ้าแบบอัตโนมัติโดยการใช้โดรน
บินเก็บภาพพื้นที่ที่ยังไม่มีไฟฟ้า (สภาพภูมิประเทศทั่วไป) กับสภาพของพื้นที่ที่เป็นภาพไฟป่าที่แสดง
ในภาพที่ 3.1 ก. และ ข. ตามลำดับ โดยใช้โดรนบินสำรวจพื้นที่เพื่อเก็บภาพพื้นที่ตัวอย่างแล้วทำการ
ตัดแยกภาพที่เป็นภาพที่เกิดไฟไหม้ และไม่เป็นไฟไหม้ จากการตัดแยกภาพตัวอย่างที่ส่งจากโดรนที่ใช้
มุมกล้อง 90 องศาเท่านั้นจึงได้ภาพที่เป็นภาพที่มีไฟไหม้ จำนวน 177 ภาพ และภาพที่ไม่มีไฟไหม้
จำนวน 177 ภาพ จากนั้นทำการตัดแยกภาพที่มีไฟไหม้ชัดเจน 100 ภาพ แล้วนำภาพที่ถูกตัดแยก
ดังกล่าวจำนวน 90 ภาพไปใช้ในกระบวนการสร้างโมเดลโดยใช้ TensorFlow เพื่อเทรนให้ระบบ
เรียนรู้เกี่ยวกับภาพว่าเมื่อโดรนขึ้นบินแล้วภาพที่ส่งมาจะเป็นภาพที่เกิดไฟป่าหรือไม่



(ก)

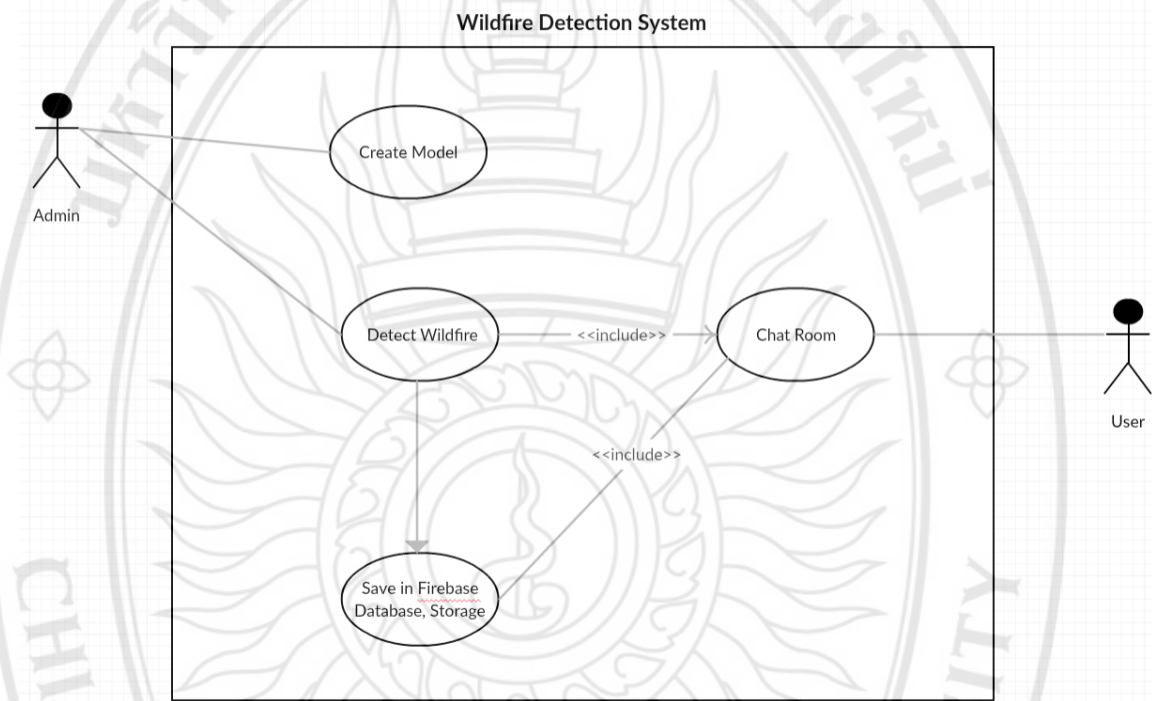


(ข)

ภาพที่ 3.1 แสดงภาพจากโดรน (ก) ภาพที่มีไฟไหม้ และ (ข) ภาพไม่มีไฟไหม้

3.2 ขั้นตอนการวิเคราะห์และออกแบบระบบการตรวจสอบไฟป่า

ขั้นตอนการวิเคราะห์และออกแบบระบบการตรวจสอบไฟป่าแบ่งบุคคลที่เกี่ยวข้อง 2 กลุ่ม ได้แก่ ผู้ดูแลระบบ และผู้ใช้ ในการบินตรวจสอบไฟป่าของโดรนได้มีการออกแบบโมเดลใหม่สำหรับตรวจสอบไฟป่า นำโมเดลไปสร้างระบบเพื่อตรวจสอบภาพที่ได้จากโดรนบินว่าเป็นภาพที่เกิดไฟป่า เมื่อตรวจสอบพบนำภาพและพิกัดส่งไปเก็บใน Firebase แล้วส่งการแจ้งเตือนการเกิดไฟป่าให้กับผู้ใช้ ในห้องแชตดัง Use Case Diagram รูปที่ 3.2



ภาพที่ 3.2 Use Case Diagrams ของระบบตรวจสอบการเกิดไฟป่า

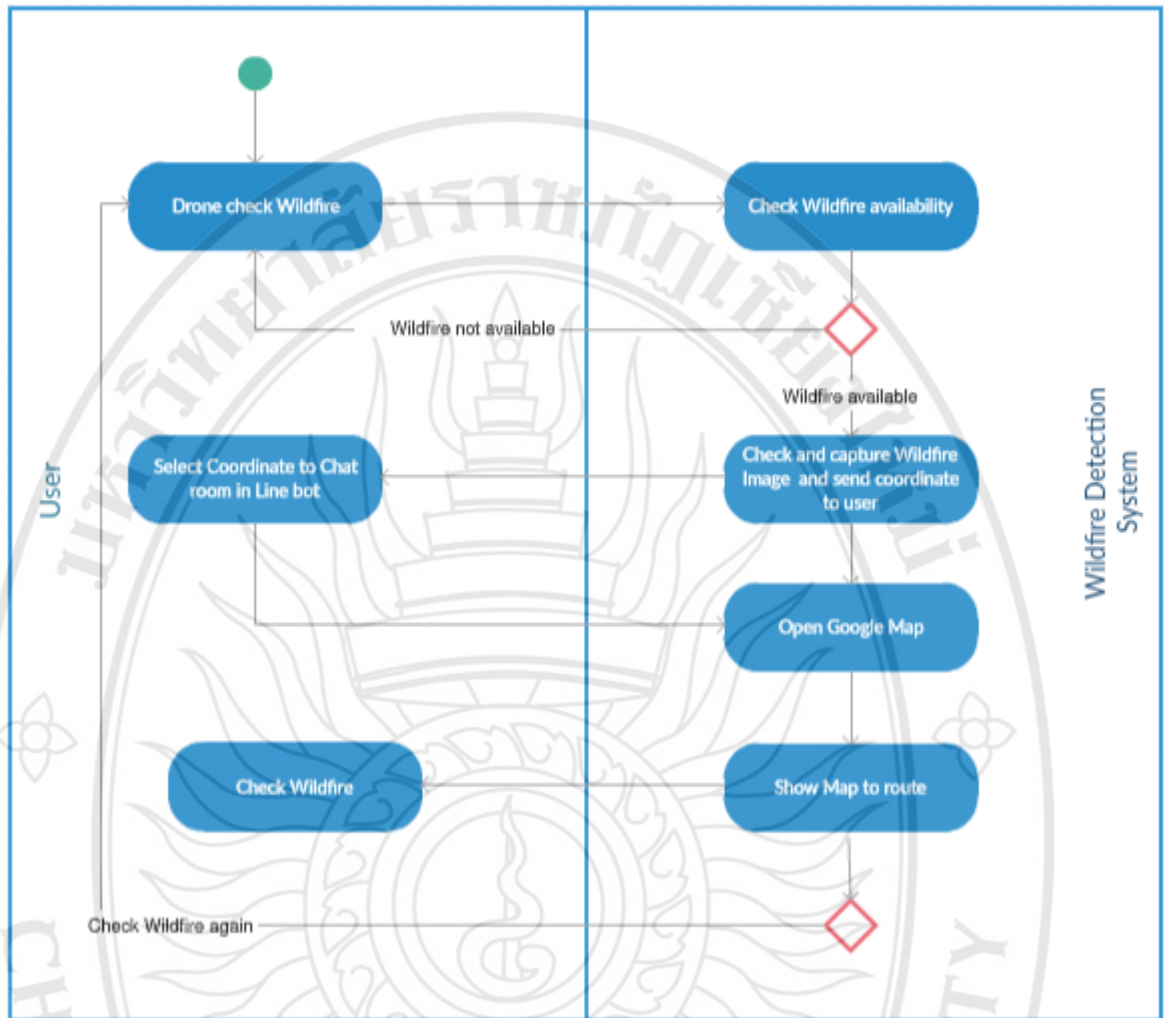
Use Case Title: Create Model	Use Case Id : 1
Primary Actor : Admin	
Stakeholder Actor : -	
Main Flow: ผู้ดูแลระบบจะทำการสร้างโมเดลใหม่ที่ใช้ในการตรวจสอบไฟป่าโดยใช้ภาพที่เก็บรวบรวมได้จากโดรนเทรนให้โมเดลเรียนรู้จากภาพจนได้โมเดลใหม่ที่สามารถนำไปใช้วิเคราะห์ภาพไฟป่าได้ในลำดับต่อไป	

Use Case Title: Detect Wildfire	Use Case Id : 2
Primary Actor : Admin	
Stakeholder Actor : User	
Main Flow: <p>ผู้ดูแลระบบสั่งให้โดรนบินเพื่อสำรวจพื้นที่ที่เกิดไฟป่าเมื่อโดรนบินสำรวจพบภาพที่น่าจะเกิดไฟป่า ระบบตรวจสอบไฟป่าจะทำการส่งการแจ้งเตือนผู้ใช้ระบบผ่านห้องแชท โดยระบบได้ส่งภาพไฟป่าและพิกัดที่ตรวจพบส่งให้กับผู้ใช้ต่อไป โดยพิกัดที่ได้จะอยู่ในแผนที่กูเกิลให้สามารถเห็นพิกัดได้ชัดเจน</p>	

Use Case Title: Save in Firebase, Storage	Use Case Id : 3
Primary Actor : Admin	
Stakeholder Actor : -	
Main Flow: <p>ผู้ดูแลระบบตรวจพบไฟป่าทำการส่งภาพไฟป่าและพิกัดไปเก็บลงใน Firebase Database</p>	

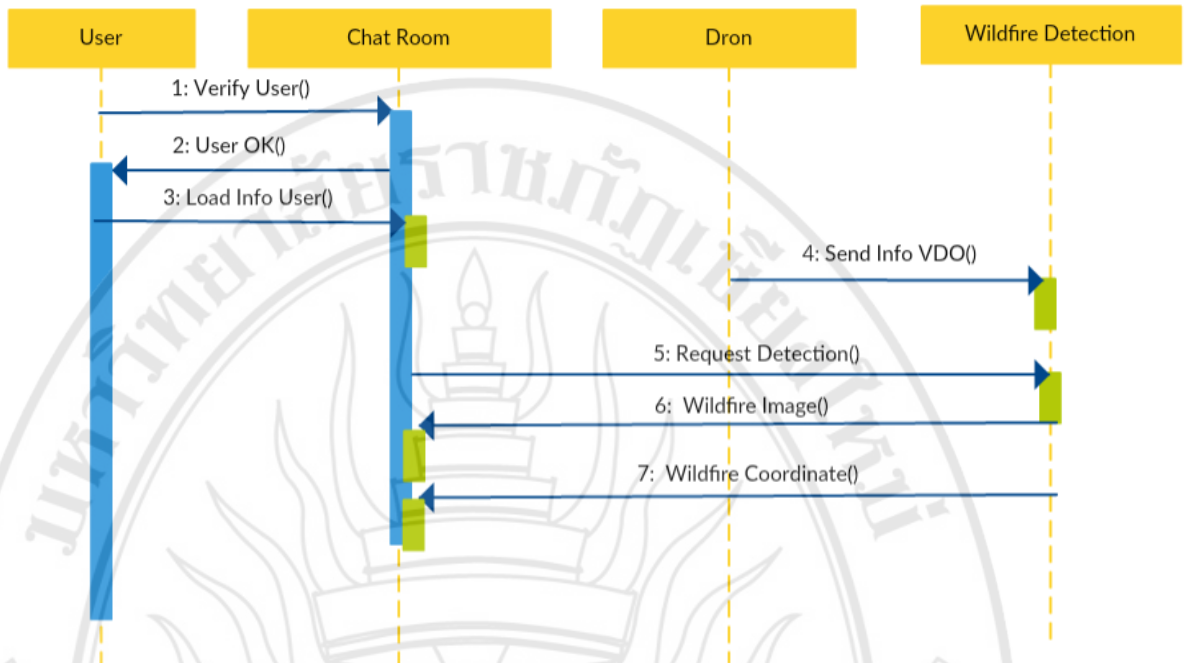
Use Case Title: Chat Room	Use Case Id : 4
Primary Actor : User	
Stakeholder Actor : -	
Main Flow: <p>ผู้ใช้จะได้รับการเพิ่มข้อมูลผู้ใช้เข้าไปในห้องแชทที่ใช้ในการส่งการแจ้งเตือนเมื่อโดรนบินแล้วระบบตรวจสอบพบไฟป่าจะทำการส่งภาพไฟป่าและพิกัดแจ้งเตือนให้ผู้ใช้ในห้องแชท</p>	

ในภาพที่ 3.3 Activity Diagrams ของระบบตรวจสอบการเกิดไฟป่าเป็นแผนภาพแสดงกิจกรรมระหว่างผู้ใช้ระบบและระบบตรวจสอบไฟป่า ผู้ใช้ให้โดรนบินสำรวจบินป่าระบบทำการตรวจสอบภาพที่ได้จากโดรนเมื่อตรวจไม่พบก็ทำการบินสำรวจไปเรื่อยๆ จนกว่าจะตรวจพบไฟป่าถึงทำการส่งภาพไฟป่าและพิกัดให้ผู้ใช้ผ่านห้องแชทในไลน์โดยผู้ใช้สามารถเปิดแผนที่ตรวจสอบตำแหน่งไฟป่าได้จากแผนที่กูเกิลที่สามารถแสดงแผนที่และเส้นทางไปตรวจสอบไฟป่าได้



ภาพที่ 3.3 Activity Diagrams ของระบบตรวจสอบการเกิดไฟป่า

ในการแสดง Sequence Diagrams ของระบบตรวจสอบการเกิดไฟป่าดังภาพที่ 3.4 ผู้ดูแลระบบทำการเพิ่มผู้ใช้เข้าห้องแชทไลน์ของการตรวจสอบไฟป่าในชุมชน โดรนทำการบินแล้วตรวจสอบพบไฟป่า ระบบจะทำการส่งภาพการเกิดไฟป่าที่ตรวจพบจะส่งการแจ้งเตือนด้วยภาพ และพิกัดของสถานที่ที่เกิดไฟป่าเข้ามายังห้องแชท



ภาพที่ 3.4 Sequence Diagrams ของระบบตรวจสอบการเกิดไฟป่า

3.3 วิธีการสร้างโมเดลตรวจสอบไฟป่า

วิธีการสร้างโมเดลตรวจสอบไฟป่าด้วยไลบรารีของ TensorFlow ต้องมีวิธีการสร้างโมเดลตรวจสอบไฟป่าดังต่อไปนี้

1) ทำการติดตั้งโปรแกรม Docker เพื่อจำลองสภาพแวดล้อมเครื่องแม่ข่ายขึ้นมาใช้ในการ Run Service ที่ต้องการ นั่นคือจะใช้ Container ในการจำลองสภาพแวดล้อมขึ้นมาเพื่อใช้งานสำหรับ 1 service ที่ต้องการใช้งานเท่านั้น

2) รัน Docker Container เพื่อเทรนโมเดล (Train Model) สำหรับตรวจสอบไฟป่า ข้อสังเกต ต้องมีการติดตั้ง Docker

3) รัน Docker เพื่อเริ่มใช้งาน Docker Container มีการแชร์โฟลเดอร์อยู่ที่ android_tensorflow_wildfire_detection ให้มองเห็นร่วมกันระหว่าง tf_files:/tf_files ที่อยู่ใน docker container และข้างนอกคือเครื่องคอมพิวเตอร์ของผู้ใช้ ดังคำสั่งด้านล่างนี้

```
docker run -it -v
$HOME/android_tensorflow_wildfire_detection/tf_files:/tf_files
danjarvis/tensorflow-android:1.0.0
```

4) เริ่มเทรนโมเดลบนไลบรารีของ tensorflow ด้วยการตรวจสอบภาพที่ทำการคัดแยกไว้ในโฟลเดอร์ tf_files ซึ่งต้องทำการเตรียมเก็บรวบรวมภาพไฟป่าเพื่อนำไปเก็บไว้ใน /tf_files/image/Fire จำนวน 177 ภาพ และภาพที่ไม่มีไฟป่าไปเก็บไว้ใน /tf_files/image/None จำนวน 177 ภาพ โดยไฟล์ภาพต้องทำการแปลงไฟล์เป็น JPG แล้วจากนั้นทำการสร้างไฟล์ retrained_labels.txt จากนั้นทำการรันคำสั่งดังด้านล่างนี้

```
cd /tensorflow
```

```
python tensorflow/examples/image_retraining/retrain.py \  
--bottleneck_dir=/tf_files/bottlenecks \  
--how_many_training_steps 500 \  
--model_dir=/tf_files/inception \  
--output_graph=/tf_files/retrained_graph.pb \  
--output_labels=/tf_files/retrained_labels.txt \  
--image_dir /tf_files/images
```



```

root@77228b54b670:/tensorflow# python tensorflow/examples/image_retraining/retrain.py \
> --bottleneck_dir=tf_files/bottlenecks \
> --how_many_training_steps 500 \
> --model_dir=tf_files/inception \
> --output_graph=tf_files/retrained_graph.pb \
> --output_labels=tf_files/retrained_labels.txt \
> --image_dir /tf_files/images

W tensorflow/core/platform/cpu_feature_guard.cc:45] The TensorFlow library wasn't compiled to use SSE3 instructions, but these are available on your machine; using the --enable_sse3 flag could improve performance. Instead see https://magenta.tensorflow.org/sse_for_cortex-m56 for more information.
W tensorflow/core/platform/cpu_feature_guard.cc:45] The TensorFlow library wasn't compiled to use SSE4.1 instructions, but these are available on your machine; using the --enable_sse41 flag could improve performance. Instead see https://magenta.tensorflow.org/sse_for_cortex-m56 for more information.
W tensorflow/core/platform/cpu_feature_guard.cc:45] The TensorFlow library wasn't compiled to use SSE4.2 instructions, but these are available on your machine; using the --enable_sse42 flag could improve performance. Instead see https://magenta.tensorflow.org/sse_for_cortex-m56 for more information.
W tensorflow/core/platform/cpu_feature_guard.cc:45] The TensorFlow library wasn't compiled to use AVX instructions, but these are available on your machine; using the --enable_avx flag could improve performance. Instead see https://magenta.tensorflow.org/sse_for_cortex-m56 for more information.
Looking for images in 'fire'
Looking for images in 'none'
100 bottleneck files created.
200 bottleneck files created.
300 bottleneck files created.
2019-01-27 03:58:42.195169: Step 0: Train accuracy = 59.0%
2019-01-27 03:58:42.195395: Step 0: Cross entropy = 0.635091
2019-01-27 03:58:42.684793: Step 0: Validation accuracy = 51.0% (N=100)
2019-01-27 03:58:47.731798: Step 10: Train accuracy = 95.0%
2019-01-27 03:58:47.731917: Step 10: Cross entropy = 0.395001
2019-01-27 03:58:48.213380: Step 10: Validation accuracy = 100.0% (N=100)
2019-01-27 03:58:53.569714: Step 20: Train accuracy = 92.0%
2019-01-27 03:58:53.569851: Step 20: Cross entropy = 0.333366
2019-01-27 03:58:54.124919: Step 20: Validation accuracy = 98.0% (N=100)
2019-01-27 03:58:59.868780: Step 30: Train accuracy = 98.0%
2019-01-27 03:58:59.868951: Step 30: Cross entropy = 0.241551
2019-01-27 03:59:00.839403: Step 30: Validation accuracy = 100.0% (N=100)
2019-01-27 03:59:07.306786: Step 40: Train accuracy = 97.0%
2019-01-27 03:59:07.306912: Step 40: Cross entropy = 0.218886
2019-01-27 03:59:07.964877: Step 40: Validation accuracy = 100.0% (N=100)
2019-01-27 03:59:13.907290: Step 50: Train accuracy = 93.0%
2019-01-27 03:59:13.907475: Step 50: Cross entropy = 0.241287
2019-01-27 03:59:14.555152: Step 50: Validation accuracy = 100.0% (N=100)
2019-01-27 03:59:19.838441: Step 60: Train accuracy = 93.0%
2019-01-27 03:59:19.838715: Step 60: Cross entropy = 0.221765
2019-01-27 03:59:20.313331: Step 60: Validation accuracy = 97.0% (N=100)
2019-01-27 03:59:25.472288: Step 70: Train accuracy = 95.0%
2019-01-27 03:59:25.472444: Step 70: Cross entropy = 0.163148
2019-01-27 03:59:25.991365: Step 70: Validation accuracy = 100.0% (N=100)
2019-01-27 03:59:31.508453: Step 80: Train accuracy = 99.0%
2019-01-27 03:59:31.508578: Step 80: Cross entropy = 0.131785
2019-01-27 03:59:31.970141: Step 80: Validation accuracy = 100.0% (N=100)

```



```

retrained_labels.txt
fire
none

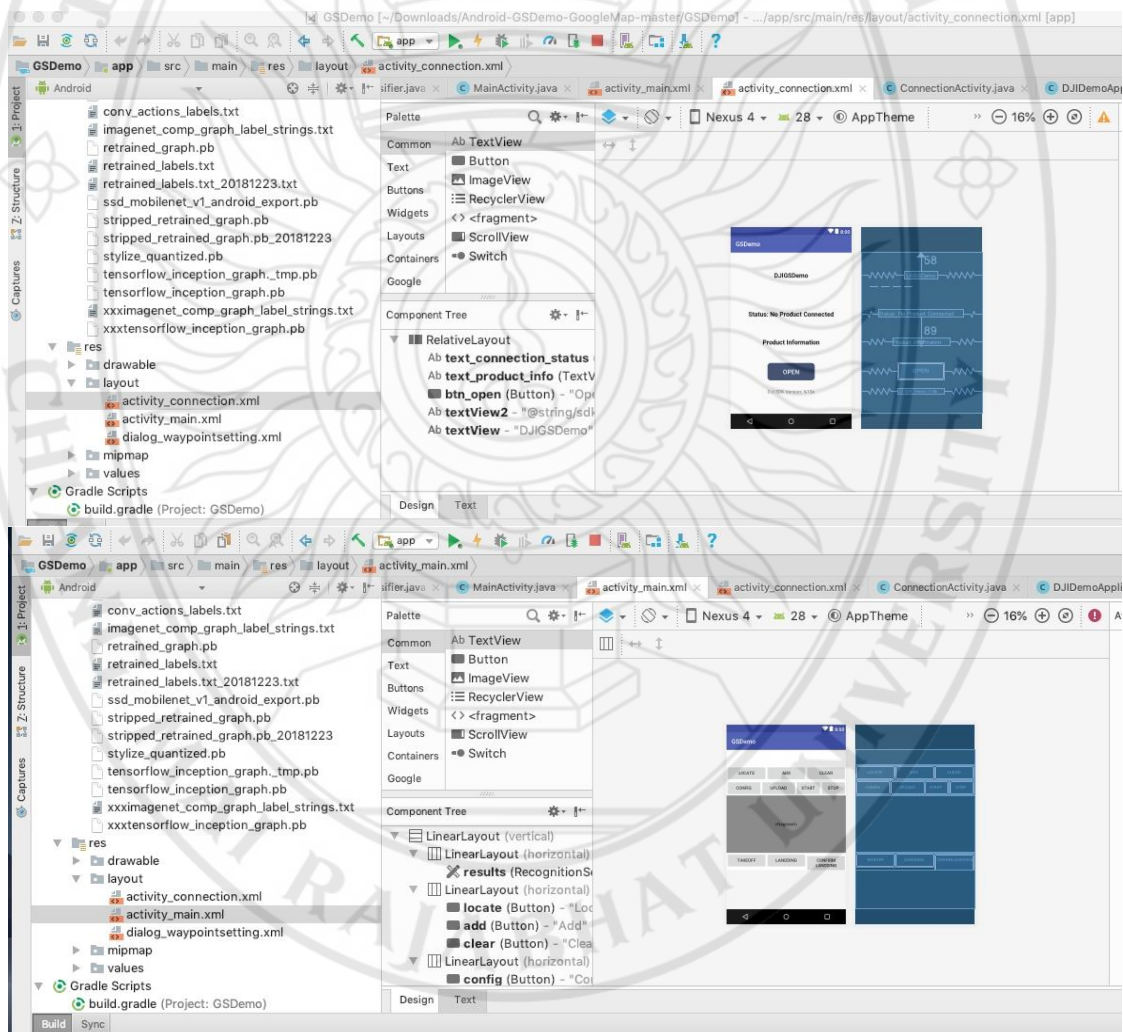
```

ภาพที่ 3.5 หน้าจอแสดงการสร้างโมเดลใหม่ที่ใช้ในการตรวจไฟฟ้า

5) เมื่อทำการรันคำสั่งแล้วจะได้ผลลัพธ์ดังภาพที่ 3.5 เพื่อเทรนโมเดลให้เรียนรู้เกี่ยวกับภาพไฟฟ้าที่ต้องการ โดยคำสั่งในการเทรนให้โมเดลรู้จักภาพที่มีไฟและไม่มีไฟแล้วก็ทำการสร้างโมเดลเก็บไว้ในไฟล์ `retrained_graph.pb` ซึ่งเป็นโมเดลที่เรียนรู้ภาพเกี่ยวกับกับการใช้ตรวจสอบไฟเรียบร้อยแล้ว

3.4 การออกแบบหน้าจอแอปพลิเคชันควบคุมโดรนเพื่อตรวจสอบไฟฟ้า

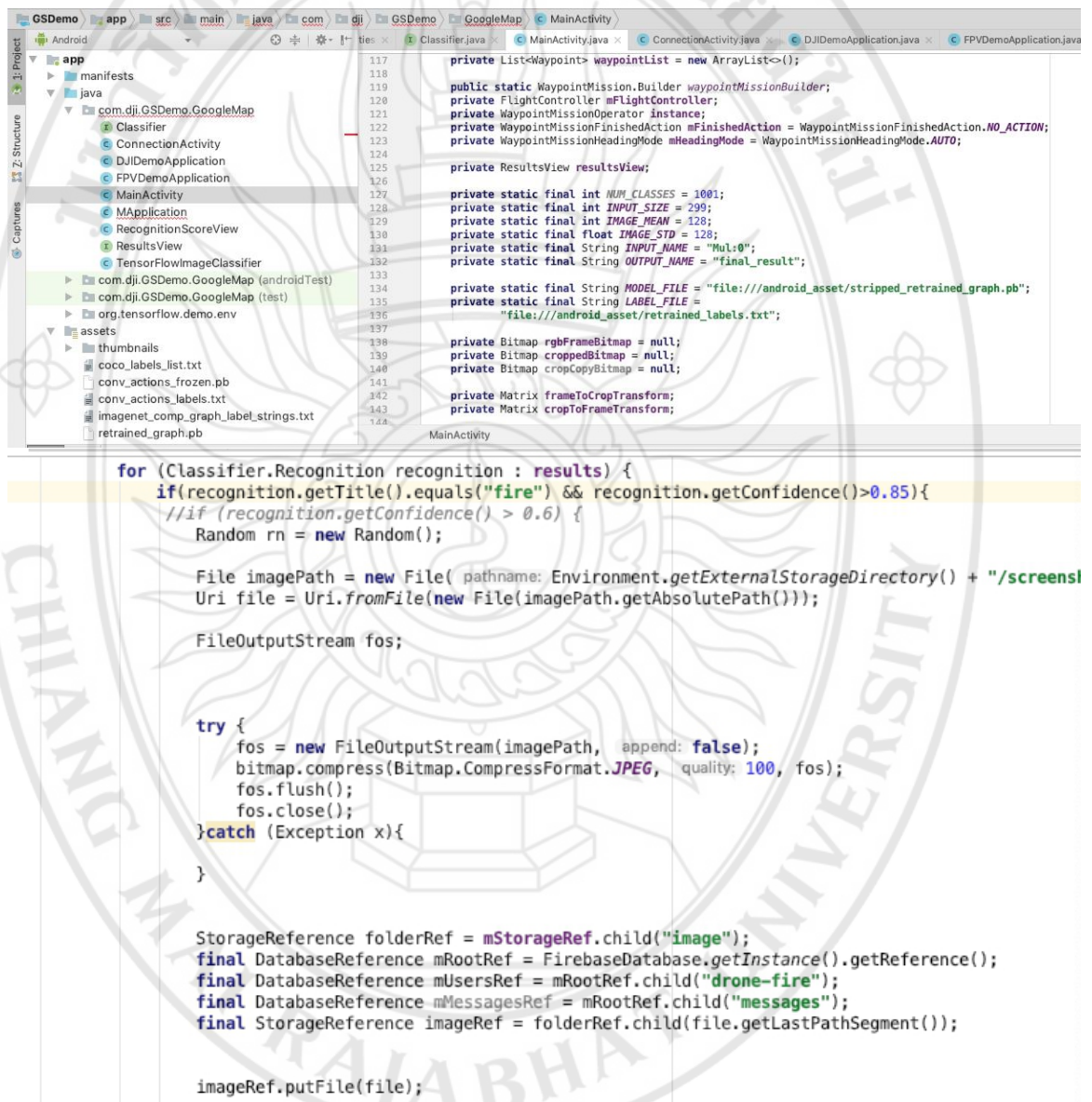
การออกแบบหน้าจอแอปพลิเคชันควบคุมโดรนเพื่อตรวจสอบไฟฟ้าโดยใช้โปรแกรม Android Studio ในการออกแบบหน้าจอการใช้งานเพื่อควบคุมโดรน ซึ่งโดรนสามารถบินอัตโนมัติด้วยฟังก์ชัน Waypoint ดังรายละเอียดภาพที่ 3.6



ภาพที่ 3.6 หน้าจอการออกแบบแอปพลิเคชันควบคุมโดรนเพื่อตรวจสอบไฟฟ้า

3.5 วิธีการนำโมเดลที่ได้จากการเทรนให้รู้จักภาพไฟมาใส่แอนดรอยด์แอปพลิเคชัน

วิธีการนำโมเดลที่ได้จากการเทรนให้รู้จักภาพไฟมาใส่แอนดรอยด์แอปพลิเคชันเพื่อนำไปประยุกต์ใช้ตรวจสอบภาพที่ส่งมาจากโดรน โดยให้ไลบรารีของ TensorFlow บน Android เข้ามา Map ข้อมูลภาพจากโดรนกับโมเดลที่สร้างขึ้นมาในไฟล์ `retrained_graph.pb` และ `retrained_labels.txt`



```

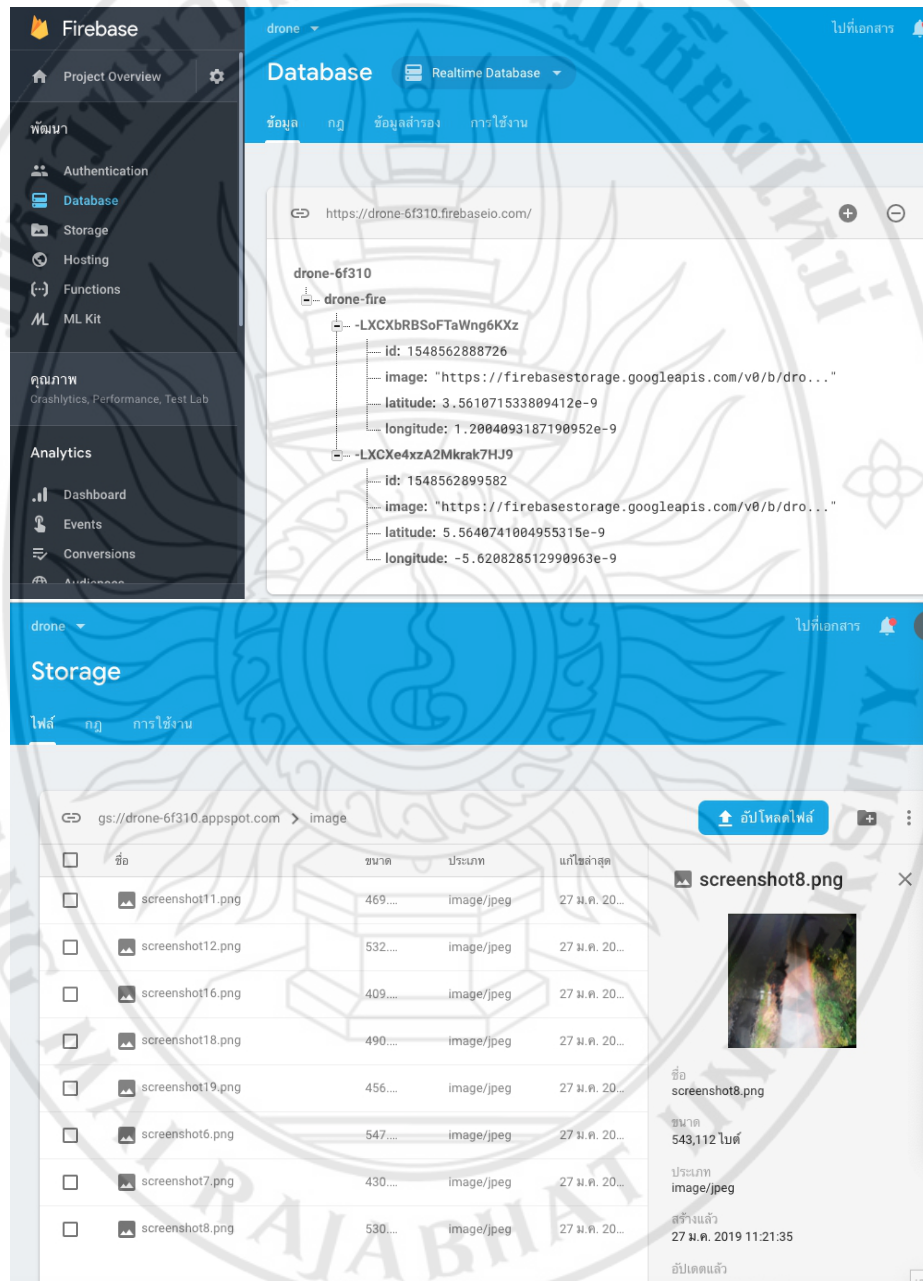
117 private List<Waypoint> waypointList = new ArrayList<>();
118
119 public static WaypointMission.Builder waypointMissionBuilder;
120 private FlightController mFlightController;
121 private WaypointMissionOperator instance;
122 private WaypointMissionFinishedAction mFinishedAction = WaypointMissionFinishedAction.NO_ACTION;
123 private WaypointMissionHeadingMode mHeadingMode = WaypointMissionHeadingMode.AUTO;
124
125 private ResultsView resultsView;
126
127 private static final int NUM_CLASSES = 1001;
128 private static final int INPUT_SIZE = 299;
129 private static final int IMAGE_MEAN = 128;
130 private static final float IMAGE_STD = 128;
131 private static final String INPUT_NAME = "Mul:0";
132 private static final String OUTPUT_NAME = "final_result";
133
134 private static final String MODEL_FILE = "file:///android_asset/stripped_retrained_graph.pb";
135 private static final String LABEL_FILE =
136     "file:///android_asset/retrained_labels.txt";
137
138 private Bitmap rgbFrameBitmap = null;
139 private Bitmap croppedBitmap = null;
140 private Bitmap cropCopyBitmap = null;
141
142 private Matrix frameToCropTransform;
143 private Matrix cropToFrameTransform;
144
145 MainActivity
146
147 for (Classifier.Recognition recognition : results) {
148     if (recognition.getTitle().equals("fire") && recognition.getConfidence() > 0.85) {
149         //if (recognition.getConfidence() > 0.6) {
150             Random rn = new Random();
151
152             File imagePath = new File( Environment.getExternalStorageDirectory() + "/screenshots");
153             Uri file = Uri.fromFile(new File(imagePath.getAbsolutePath()));
154
155             FileOutputStream fos;
156
157             try {
158                 fos = new FileOutputStream(imagePath, append: false);
159                 bitmap.compress(Bitmap.CompressFormat.JPEG, quality: 100, fos);
160                 fos.flush();
161                 fos.close();
162             } catch (Exception x){
163
164             }
165
166             StorageReference folderRef = mStorageRef.child("image");
167             final DatabaseReference mRootRef = FirebaseDatabase.getInstance().getReference();
168             final DatabaseReference mUsersRef = mRootRef.child("drone-fire");
169             final DatabaseReference mMessagesRef = mRootRef.child("messages");
170             final StorageReference imageRef = folderRef.child(file.getLastPathSegment());
171
172             imageRef.putFile(file);
173         }
174     }
175 }

```

ภาพที่ 3.7 แสดงเชื่อมโยงกันระหว่างโมเดลและแอปพลิเคชัน

3.6 วิธีการเก็บข้อมูลภาพในฐานข้อมูล

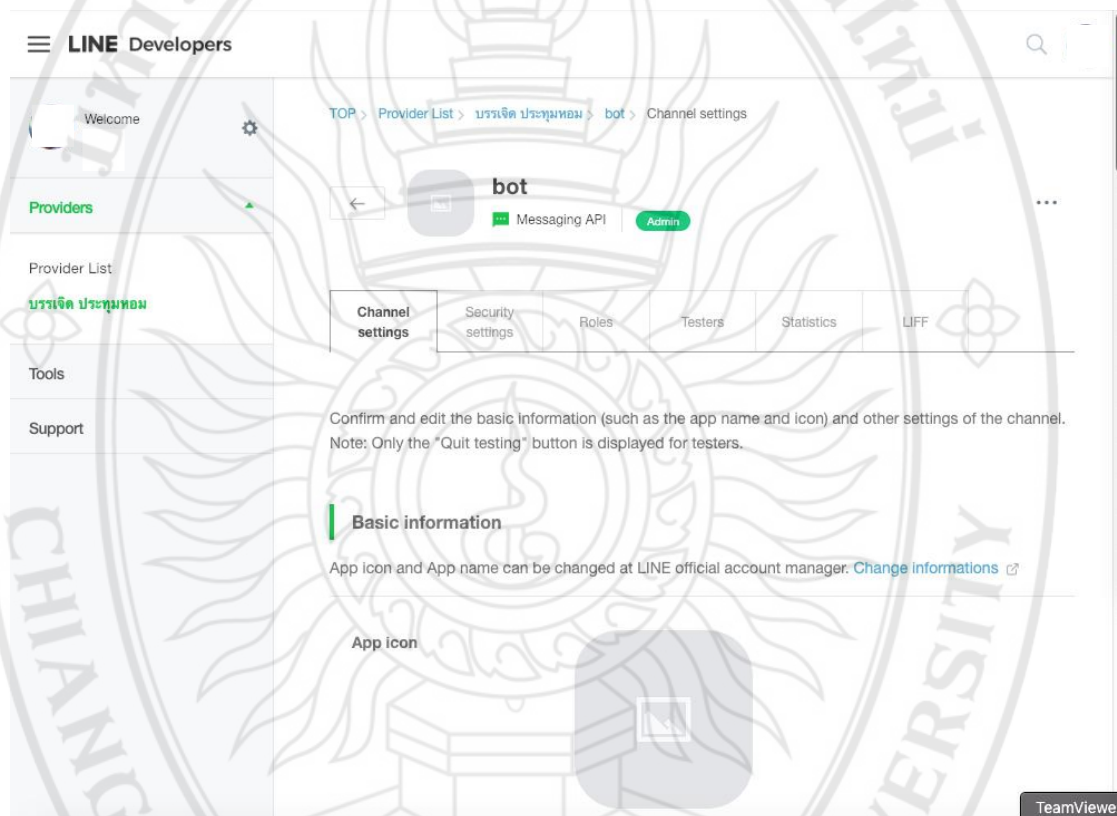
เมื่อแอปพลิเคชันตรวจสอบพบไฟฟ้าแล้วจะทำการเก็บภาพที่เป็นภาพไฟฟ้าบนทึกลงในฐานข้อมูลของ Firebase Database โดยมีฐานข้อมูลชื่อ Drone-fire ซึ่งจะมีโครงสร้างในการเก็บข้อมูลดังนี้ Id, Image, latitude และ longitude



ภาพที่ 3.8 โครงสร้างที่ใช้ในการเก็บข้อมูลใน Firebase Database

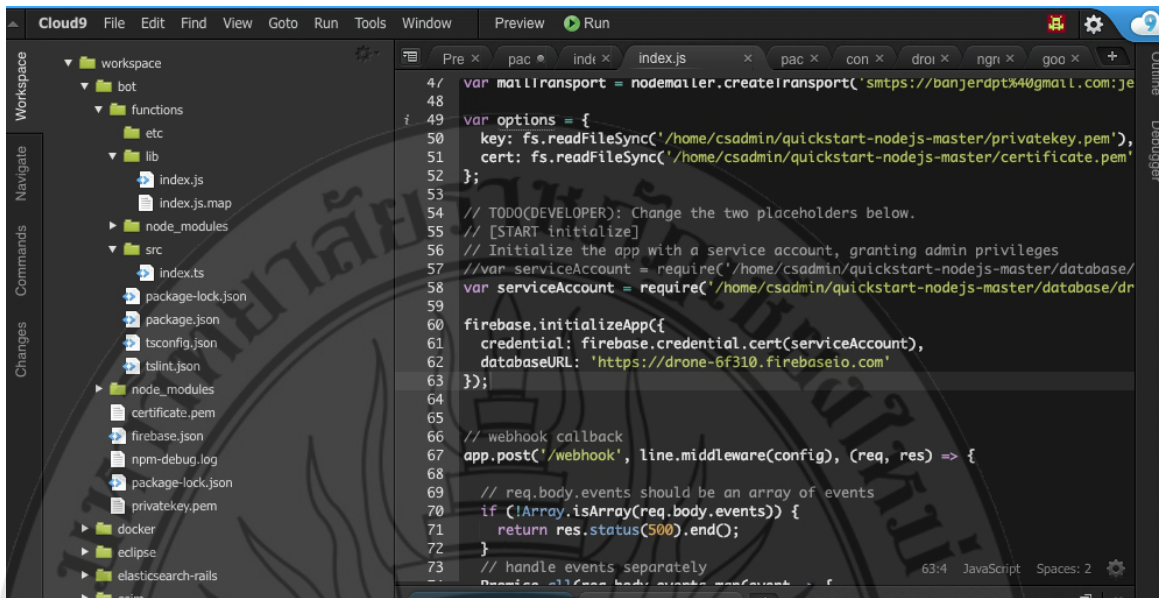
3.7 การสร้างระบบแจ้งเตือน

การสร้างระบบแจ้งเตือนด้วย Line bot ต้องทำการสมัครเป็น LINE Developers เพื่อขอ Key ดังภาพที่ 3.9 เพื่อให้ Line bot สามารถเพิ่มผู้ใช้งานเข้าไปในแชทกลุ่ม เมื่อมีการตรวจสอบพบไฟป่าจะมีการส่งภาพ และพิกัดให้กับผู้ใช้ที่เป็นสมาชิกในแชทกลุ่มเพื่อทำการแจ้งเตือนการเกิดไฟป่า



ภาพที่ 3.9 Line Developers

เมื่อทำการสมัครเป็น LINE Developers เรียบร้อยแล้วทำการเขียน NodeJS เพื่อรับข้อมูลจาก Firebase Database เมื่อมีการส่งข้อมูลมาจากโดรนว่าพบไฟป่าจะเขียนคำสั่งให้ทำการอ่านข้อมูลที่ส่งมาเก็บไว้ในโดรน ดังภาพที่ 3.10 แล้วเขียนคำสั่งเพื่อส่งข้อมูลที่มีใน Firebase Database ดังภาพที่ 3.11 เพื่อทำการส่งพิกัดและภาพเพื่อแจ้งเตือนไปยังไลน์กลุ่มของผู้ใช้งาน

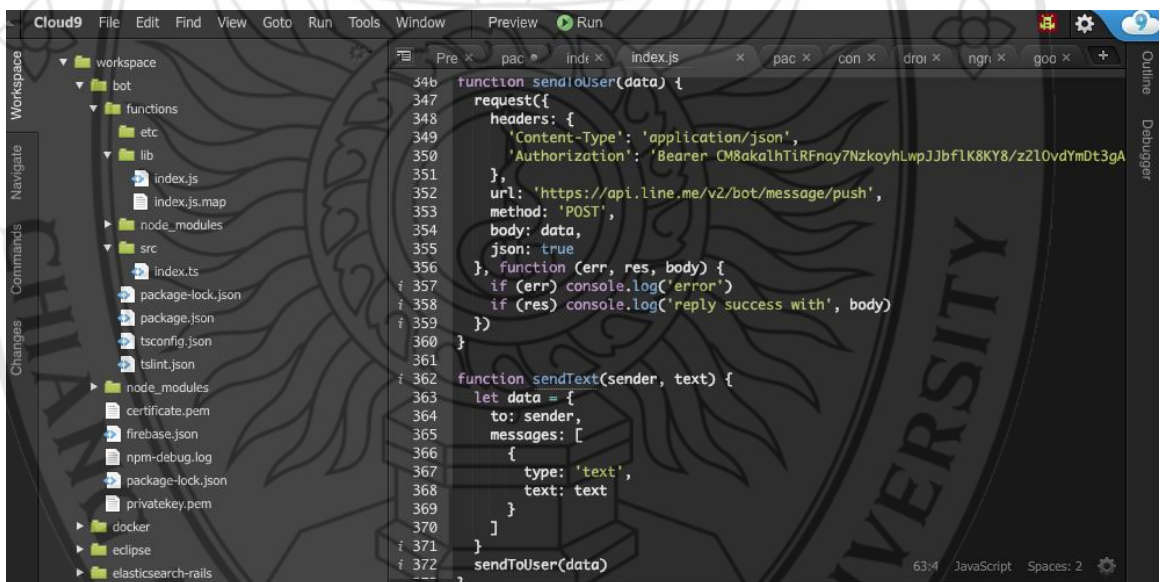


```

47 var mailTransport = nodemailer.createTransport({ smtps://banjerapt%4@gmail.com:je
48
49 var options = {
50   key: fs.readFileSync('/home/csadmin/quickstart-nodejs-master/privatekey.pem'),
51   cert: fs.readFileSync('/home/csadmin/quickstart-nodejs-master/certificate.pem')
52 };
53
54 // TODO(DEVELOPER): Change the two placeholders below.
55 // [START initialize]
56 // Initialize the app with a service account, granting admin privileges
57 //var serviceAccount = require('/home/csadmin/quickstart-nodejs-master/database/
58 var serviceAccount = require('/home/csadmin/quickstart-nodejs-master/database/dr
59
60 firebase.initializeApp({
61   credential: firebase.credential.cert(serviceAccount),
62   databaseURL: 'https://drone-6f310.firebaseio.com'
63 });
64
65
66 // webhook callback
67 app.post('/webhook', line.middleware(config), (req, res) => {
68
69   // req.body.events should be an array of events
70   if (!Array.isArray(req.body.events)) {
71     return res.status(500).end();
72   }
73   // handle events separately
74   req.body.events.forEach(event => {

```

ภาพที่ 3.10 คำสั่งในอ่านข้อมูลจาก firebase



```

346 function sendToUser(data) {
347   request({
348     headers: {
349       'Content-Type': 'application/json',
350       'Authorization': 'Bearer CM8akalhtIRFnay7NzkoyhLwpJJbFLK8KY8/z210vdYmDt3gA
351     },
352     url: 'https://api.line.me/v2/bot/message/push',
353     method: 'POST',
354     body: data,
355     json: true
356   }, function (err, res, body) {
357     if (err) console.log('error')
358     if (res) console.log('reply success with', body)
359   })
360 }
361
362 function sendText(sender, text) {
363   let data = {
364     to: sender,
365     messages: [
366       {
367         type: 'text',
368         text: text
369       }
370     ]
371   }
372   sendToUser(data)

```

ภาพที่ 3.11 คำสั่งในส่งข้อมูลจาก firebase ไปยังไลน์กลุ่มผู้ใช้

3.8 วิธีการประเมินผลสัมฤทธิ์

การตรวจสอบเป้าหมาย (ไฟ) โดยระบบสามารถแจ้งผลการตรวจสอบว่าเป็นไฟโดยดูค่าความถูกต้องและค่าแม่นยำที่ทำการทดลองในแต่ละครั้ง

ความถูกต้อง หรือแม่นยำ (Accuracy) เป็นค่าที่บ่งบอกถึงความสามารถของเครื่องมือวัด (Instrument) ในการอ่านค่าหรือแสดงค่าที่วัดหรือตรวจสอบได้เข้าใจค่าจริง

โดยการคำนวณค่าความถูกต้อง/ความแม่นยำใช้สมการดังนี้

$$\%Accuracy = 100 - \%Error$$

โดยที่

$$\text{Relative error} = \left| \frac{X_{mea} - X_t}{X_t} \right| \quad (1)$$

$$\%Error = \text{Relative error} \times 100$$

เมื่อ X_{mea} คือค่าที่ได้จากการวัด (Measure Value)

X_t คือ ค่าจริง (True Value)