

Application of the ONVIF Protocol to support IP camera management and video streaming through websites

การประยุกต์ใช้โปรโตคอล ONVIF เพื่อรองรับการจัดการกล้องไอพีและการสตรีมวิดีโอผ่านเว็บไซต์

Topics

01

Why

อธิบายถึงที่มาและปัญหา
ของโครงการ

02

Solve

อธิบายถึงวัตถุประสงค์
ขอบเขตและประโยชน์

03

Operation

อธิบายถึงภาพรวมโครงการ
และแนวทางการพัฒนา

04

Conclude

อธิบายถึงผลการดำเนินการ
และการทดสอบ

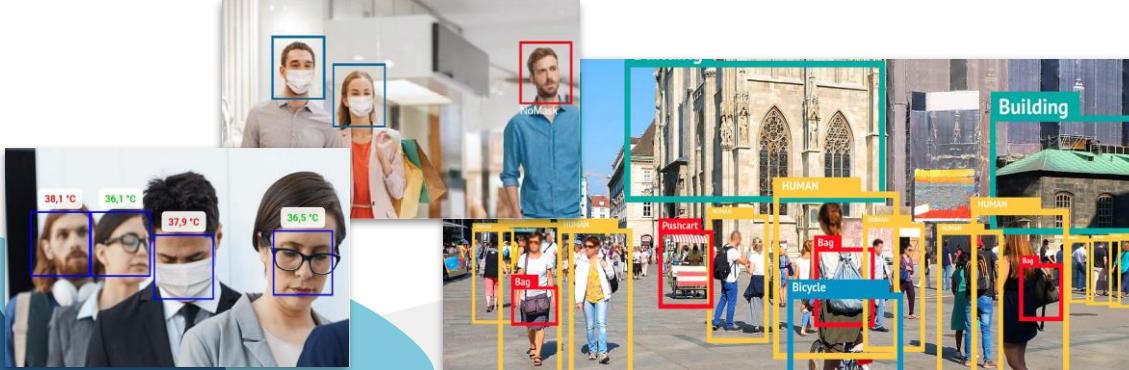
01

Why

อธิบายถึงที่มาและปัจจัยของโครงงาน

Why

- ความยุ่งยากในการใช้ CCTV แบบเก่า
- การมีผู้ผลิตหลากหลายแบรนด์
- การพัฒนาของเทคโนโลยี AI



STARCAM®

tp-link



DVR

HVR

NVR



02 Solve

อธิบายถึงวัตถุประสงค์ ขอบเขตและประโยชน์ของโครงการ

แนวทางแก้ปัญหา

วัตถุประสงค์



- เพื่อศึกษาและพัฒนาซอฟต์แวร์ที่สามารถสนับสนุนการจัดการกับกล้องไอพี ได้อย่างมีประสิทธิภาพ เช่น การเพิ่มหรือลบที่อยู่ IP ของกล้อง และการเรียกดูภาพการสตรีมจากกล้องที่ได้ถูกเพิ่มเข้าสู่ระบบ
- เพื่อสนับสนุนและวิเคราะห์ความสามารถให้ผู้ใช้สามารถแสดงภาพวิดีโอและข้อมูลเมตา จากกล้อง IP ผ่านหน้าเว็บไซต์
- เพื่อศึกษาและเข้าใจถึงกระบวนการรับส่งข้อมูลเมตาโดยใช้รูปแบบมาตรฐานของ ONVIF



IP Camera Protocol

PSIA



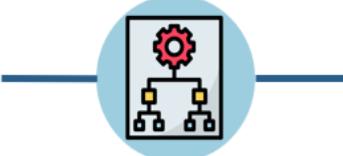
- ใช้งานได้ง่าย
- ใช้สากลเปตยกรรม (REST)
- ใช้ uPNP, Zeroconf, Bonjour

ONVIF



- ใช้ XML, SOAP, WSDL
- ใช้งานได้ดีมาก, มีความเชื่อมงวด
- การสตรีมวิดีโอ, การจัดการ PTZ
- การจัดการฟังก์ชันการทำงาน / วิเคราะห์การสตรีม

GB / T28181



- เป็น Protocol ของประเทศจีน
- มีเพื่อแก้ปัญหาการผูกขาดของ ONVIF
- แต่ยังต้องทำงานร่วมกับ ONVIF protocol อญดี



XML

กำหนดรูปแบบข้อมูล เพื่อให้แต่ละอุปกรณ์
เข้าใจและแลกเปลี่ยนข้อมูลระหว่างกันได้

ONVIF Protocol จะใช้ XML เพื่อกำหนดรูปแบบข้อมูล, ใช้ SOAP เป็นโปรโตคอลสื่อสารและ
แพ็กเกจข้อมูล XML และใช้ WSDL เป็นภาษาที่ใช้ในการบรรยายและระบุเซอร์วิสวีเบ็บ
ONVIF Protocol ความสัมพันธ์ระหว่างเทคโนโลยีเหล่านี้ช่วยให้อุปกรณ์เครือข่ายวิดีโอที่เข้าร่วมมาตรฐาน
ONVIF สามารถสื่อสารและทำงานร่วมกันได้อย่างมีประสิทธิภาพและสอดคล้องกันได้



WSDL

ใช้ในการบรรยายและระบุเซอร์วิสวีเบ็บ
เซอร์วิสที่ให้บริการ เช่น
รายละเอียดของเมธอด (methods)



SOAP

เป็นโปรโตคอลที่ใช้ในการแพ็คเกจข้อมูล
XML และส่งผ่านเครือข่าย
โดย SOAP จะเป็นตัวกลาง

OnVIF®



Profile S

- การสตรีมวิดีโอและการกำหนดค่าในการสตรีม



Profile M

- ส่งเหตุการณ์ผ่านการสตรีมข้อมูลเมตา,
ด้วย ONVIF event service หรือผ่าน MQTT
- กำหนดค่าการวิเคราะห์และการสตรีมข้อมูลเมตา
- รองรับการจำแนกวัตถุ



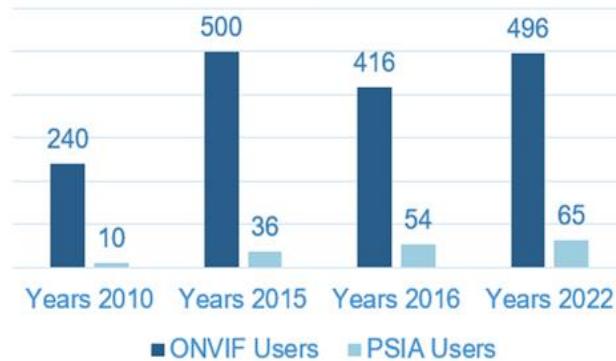
Profile T

- การสตรีมวิดีโอและข้อมูลเมตา
- รองรับการบีบอัดวิดีโอแบบ H.264 / H.265
- การตั้งค่าในการถ่ายภาพจากกล้องวิดีโอ
- แจ้งเตือนการเคลื่อนไหวและแจ้งเตือนการบุกรุก
- ระบบเสียงสองทิศทาง

Comparison Graph



Partner



Items



แนวทางแก้ปัญหา

ขอบเขตการศึกษา

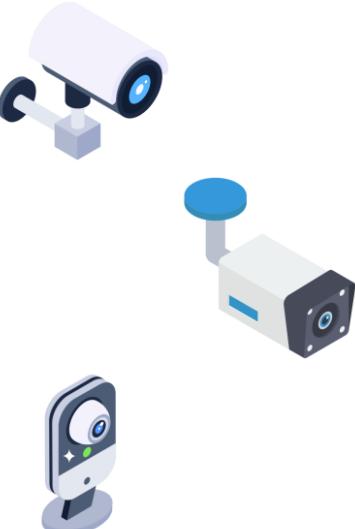
- System design : มุ่งเน้นสำหรับการจัดการกล้องไอพี
 การสตรีมวิดีโอและข้อมูลเมต้าผ่านเว็บไซต์
 - รับส่งข้อมูลภาพการสตรีม ด้วยมาตรฐาน HTTP (Hypertext Transfer Protocol)
 - รับส่งข้อมูลเมต้า ด้วยมาตรฐาน ONVIF (Open Network Video Interface Forum)
 - 1) GetDeviceInformation, 2) GetUser, 3) GetSystemDateAndTime
 - 4) GetCapabilities, 5) GetHostname, 6) GetNetworkProtocols
 - 7) GetDiscoveryMode และ 8) GetServiceCapabilities
 - สามารถจัดการ เพิ่ม-ลด เลือกดูภาพการสตรีมหรือข้อมูลกล้องไอพีได้ที่หน้าเว็บไซต์
 - ใช้อินเทอร์เฟซชั้นนำของอุปกรณ์กล้องไอพีเป็น IP Webcam
- Main Program Language : Golang
- Database : SQLite
- Web Development : HTML CSS และ JavaScript

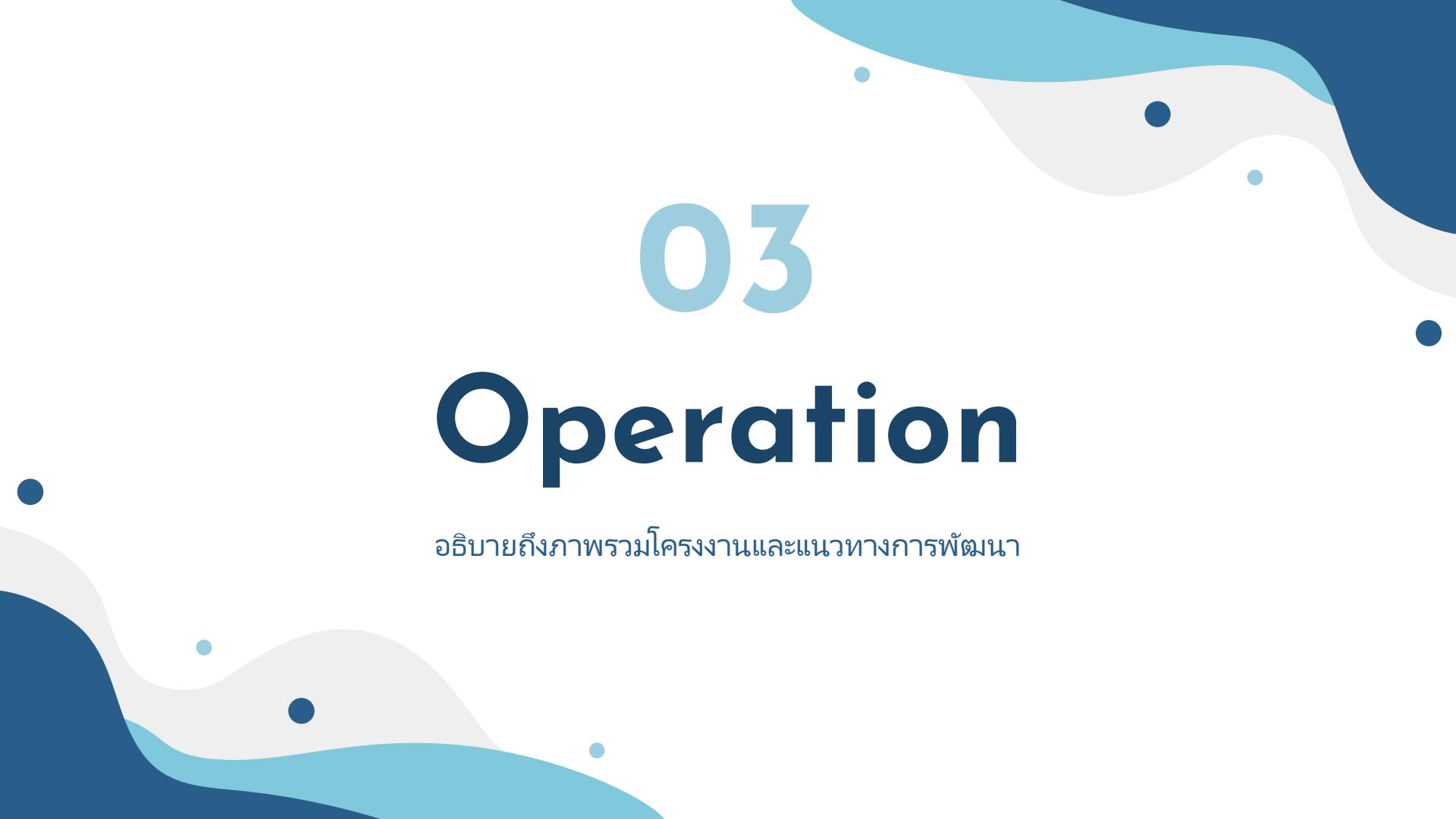


แนวทางแก้ปัญหา

ประโยชน์ที่คาดว่าจะได้รับ

- สามารถเพิ่มที่อยู่ของกล้องไอพีให้กับผู้ใช้งานที่มีกล้องไอพีมากกว่า 1 อุปกรณ์
- สามารถทำการสตรีมข้อมูลวิดีโอและข้อมูลเมตาไปยังหน้าเว็บไซต์
- ที่หน้าเว็บไซต์สามารถทำการจัดการเพิ่ม-ลดหรือเลือกดูภาพจากกล้องไอพีได้



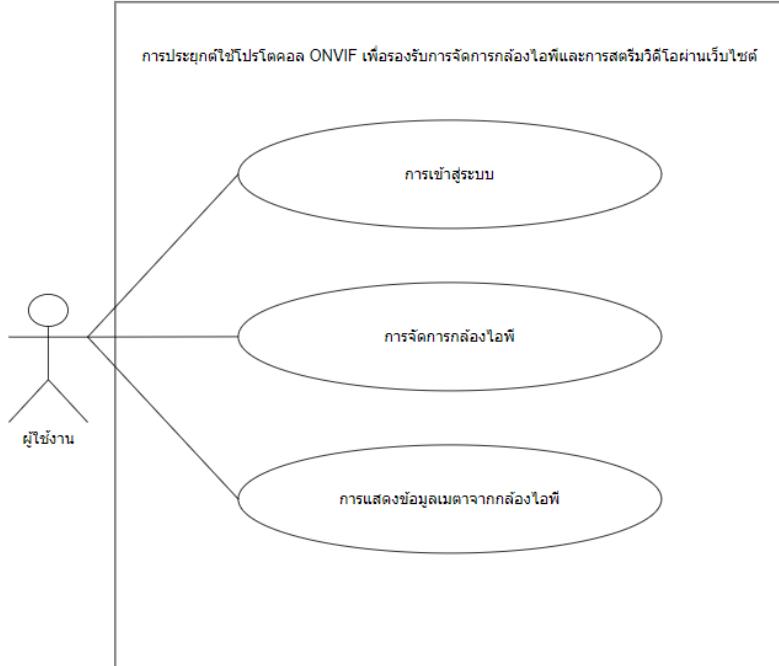


03

Operation

อธิบายถึงภาพรวมโครงงานและแนวทางการพัฒนา

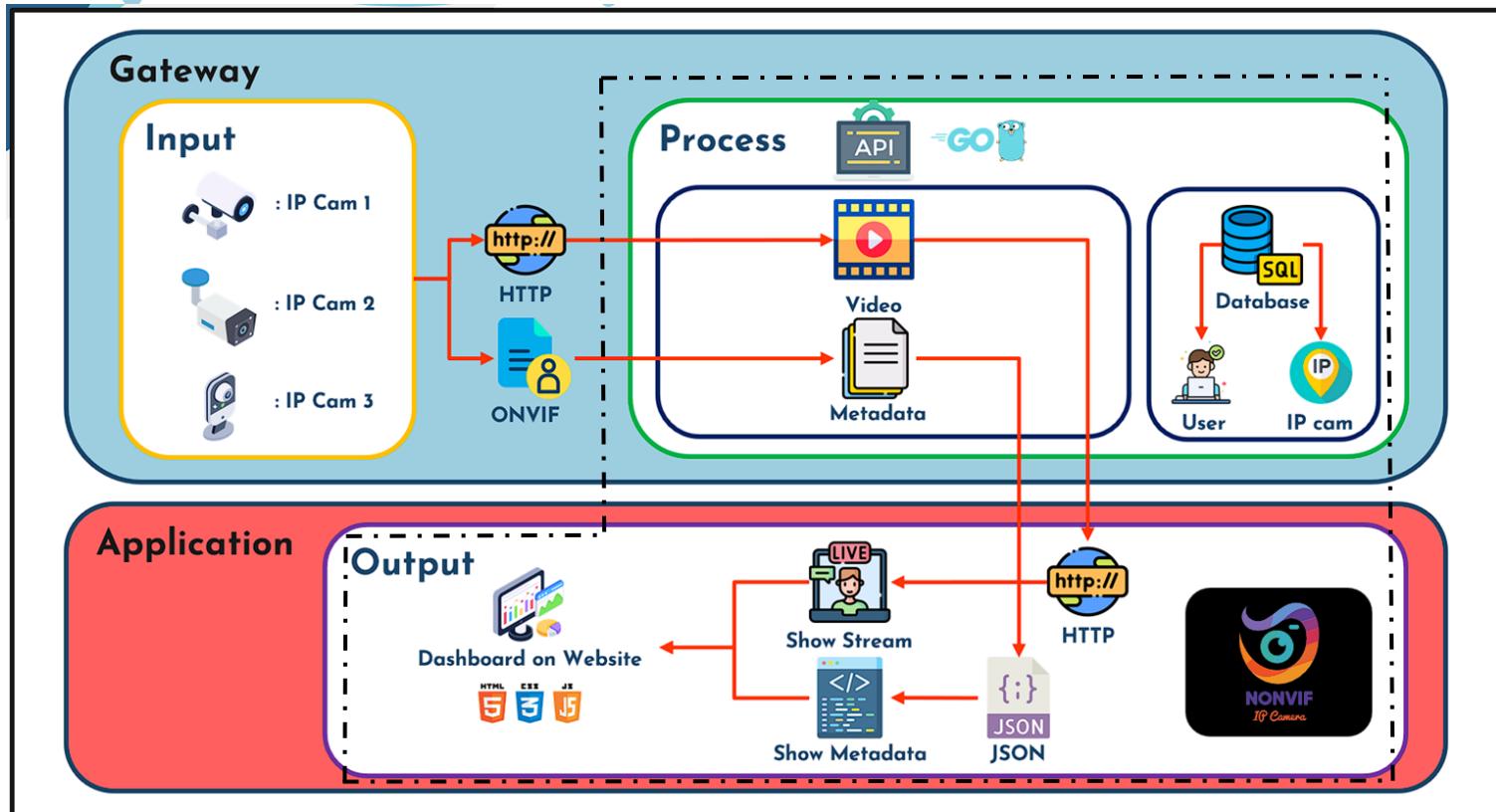
Use Case Diagram



Use Case Diagram

ลำดับ	รายการ
การเข้าสู่ระบบ	
1	ผู้ใช้งานสามารถสร้างบัญชีเพื่อใช้เข้าสู่ระบบ
2	ผู้ใช้งานสามารถเข้าสู่ระบบด้วยอีเมลและรหัสผ่าน
3	ผู้ใช้งานสามารถกดลิ้นรัหสผ่านและทำการสร้างรหัสผ่านใหม่
4	ผู้ใช้งานสามารถเลือกคูหาข้อของการสตีมภาพหรือหน้าการแสดงข้อมูลเมตตาได้
5	ผู้ใช้งานสามารถกดออกจากระบบ
การจัดการกล้องไอพี	
1	ผู้ใช้งานสามารถเพิ่มกล้องไอพี
2	ผู้ใช้งานสามารถลบกล้องไอพี
3	ผู้ใช้งานสามารถเลือกรับชมภาพการสตีมจากกล้องไอพีที่เพิ่งเข้ามาในระบบได้
การแสดงข้อมูลมาจากกล้องไอพี	
1	ผู้ใช้งานสามารถดูข้อมูลมาจากกล้องไอพีได้

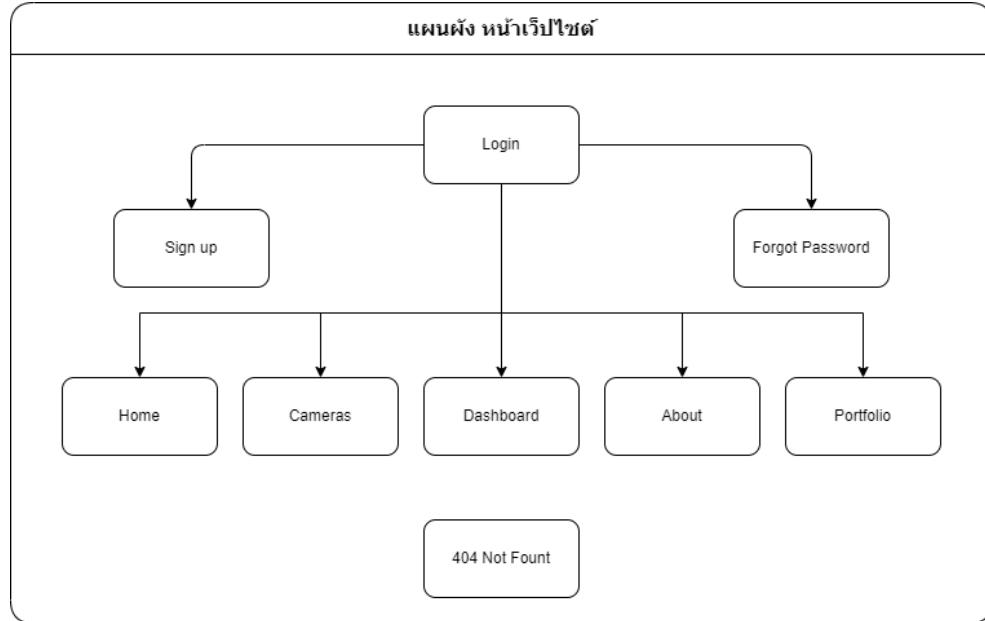
Overall Diagram



Development

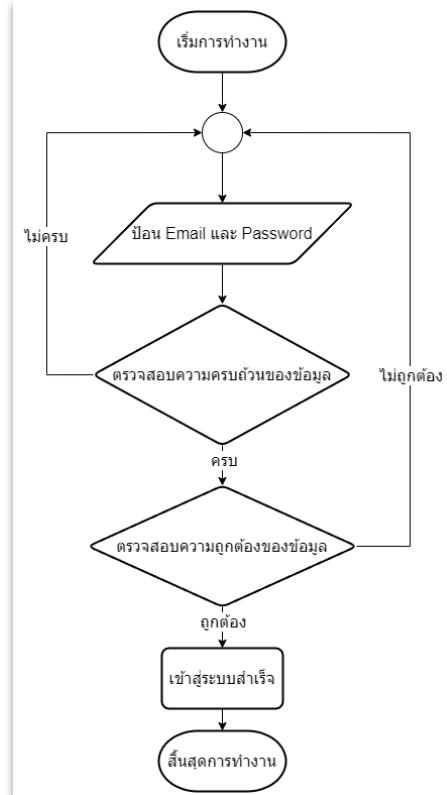
ส่วนของการพัฒนา

แผนผัง หน้าเว็บไซต์



Flow Chart

Login

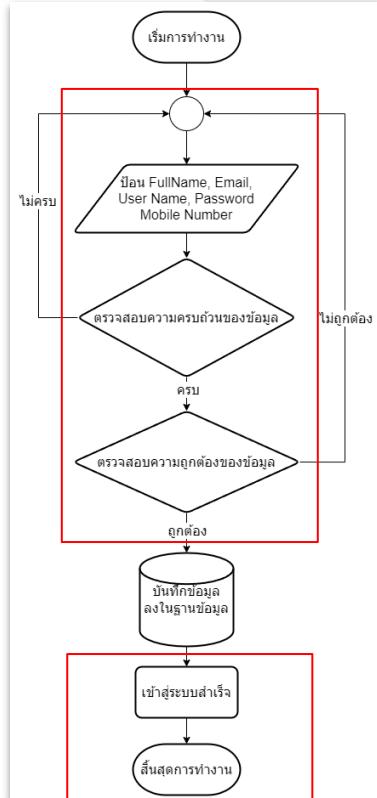


```
// API Login
// The Login function checks if a user is already Logged in and redirects them if they are
func login(w http.ResponseWriter, r *http.Request) {
    session, err := store.Get(r, "login-session")
    FetchError(err)
    ok, _ := session.Values["username"]
    if ok {
        http.Redirect(w, r, "/", http.StatusSeeOther)
        return
    }
    if !ok {
        err := loginView.Template.Execute(w, nil)
        FetchError(err)
    }
}

// API Login Authorization
// This function handles user authentication and session creation for a login page in a
func loginAuth(w http.ResponseWriter, r *http.Request) {
    r.ParseForm()
    email := r.FormValue("email")
    password := r.FormValue("password")
    user, err := GetUser(email, password)
    FetchError(err)
    if Len(user) > 0 {
        session, err := store.Get(r, "login-session")
        if err != nil {
            http.Error(w, err.Error(), http.StatusInternalServerError)
            return
        }
        session.Options = &sessions.Options{
            Path:      "/",
            MaxAge:   86400 * 30,
            HttpOnly: true,
        }
        session.Values["username"] = "username"
        err = session.Save(r, w)
        if err != nil {
            http.Error(w, err.Error(), http.StatusInternalServerError)
            return
        }
        http.Redirect(w, r, "/", http.StatusFound)
    } else {
        err := loginView.Template.Execute(w, "Please give me right email or password")
        FetchError(err)
    }
}
```

Flow Chart

Sign up



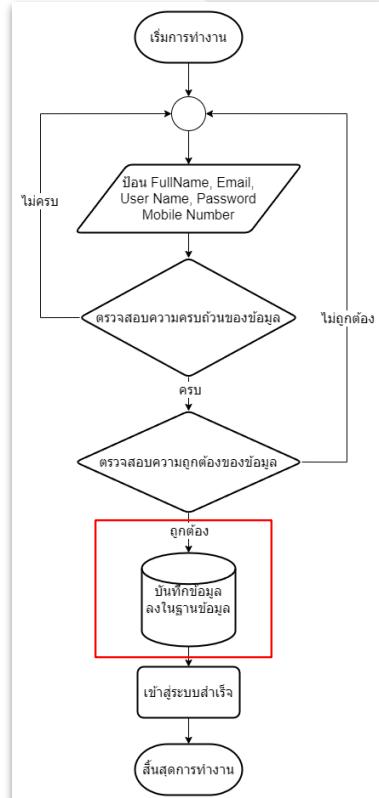
```
// API Signup page
// The function checks if a user is Logged in and redirects them to the homepage
// otherwise it displays the signup view.
func signup(w http.ResponseWriter, r *http.Request) {
    session, err := store.Get(r, "login-session")
    FetchError(err)
    _, ok := session.Values["username"]
    if !ok {
        err := signupView.Template.Execute(w, nil)
        FetchError(err)
    } else {
        http.Redirect(w, r, "/", http.StatusSeeOther)
    }
}

// API Signup Authorization page
// This function handles the signup authentication process by parsing
// and setting a session cookie before redirecting to the homepage.
func signupAuth(w http.ResponseWriter, r *http.Request) {
    r.ParseForm()
    name := r.FormValue("fullName")
    email := r.FormValue("email")
    userName := r.FormValue("userName")
    mobile := r.FormValue("mobileNumber")
    password := r.FormValue("Password")

    SignupUser(name, email, userName, mobile, password)
    session, err := store.Get(r, "login-session")
    if err != nil {
        http.Error(w, err.Error(), http.StatusInternalServerError)
        return
    }
    session.Options = &sessions.Options{
        Path:      "/",
        MaxAge:   86400 * 30,
        HttpOnly: true,
    }
    session.Values["username"] = "username"
    err = session.Save(r, w)
    if err != nil {
        http.Error(w, err.Error(), http.StatusInternalServerError)
        return
    }
    http.Redirect(w, r, "/", http.StatusFound)
}
```

Flow Chart

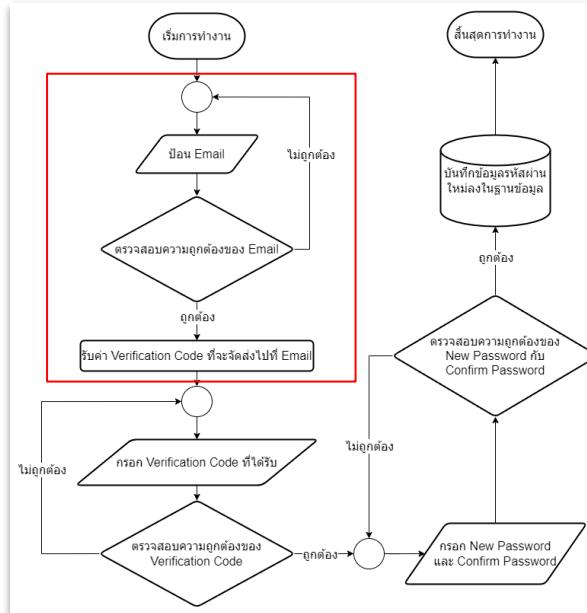
Sign up



```
// GetsignupUser
// The function inserts user data into a SQLite database table and returns the ID
func SignupUser(name, email, username, mobile, password string) (int64, error) {
    data := make(url.Values)
    data.Set("table", "user")
    data.Set("dbtype", "sqlite3")
    data.Set("Name", name)
    data.Set("Email", email)
    data.Set("UserName", username)
    data.Set("Mobile", mobile)
    data.Set("Password", password)
    id, err := msq.InsertIntoAnyTable(data, db_user)
    if err != nil {
        log.Println(err)
        return 0, err
    }
    fmt.Println("Successfully Inserted", id)
    return id, nil
}
```

Flow Chart

Forgot Password



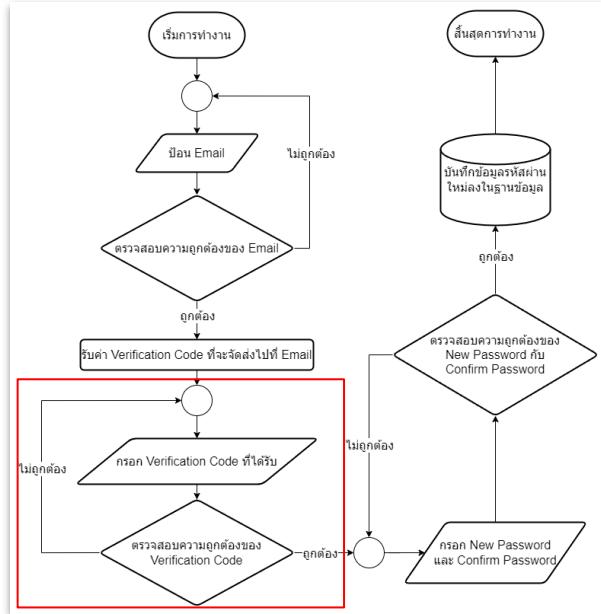
```
// API forgot Password
// The function serves the forgot password view template to the user.
func forgotPass(w http.ResponseWriter, r *http.Request) {
    err := forgotPassView.Template.Execute(w, nil)
    FetchError(err)
}

// API forgot Password Authorization
// The function handles the authentication process for a forgotten password request,
// checking if the email exists and sending a code to the user's email if it does.
var isEmail string

func forgotPassAuth(w http.ResponseWriter, r *http.Request) {
    r.ParseForm()
    email := r.FormValue("forgotEmail")
    userEmail, err := GetEmail(email)
    FetchError(err)
    if len(userEmail) > 0 {
        isEmail = userEmail[0]["Email"].(string)
        if isEmail == userEmail {
            type Data struct {
                Code string
            }
            data := Data{
                Code: strconv.Itoa(randomNUM),
            }
            err := forgotAuthView.Template.Execute(w, data)
            emailSend(isEmail)
            FetchError(err)
        } else {
            err := forgotAuthErrorView.Template.Execute(w, "Your account does not exist")
            FetchError(err)
        }
    }
}
```

Flow Chart

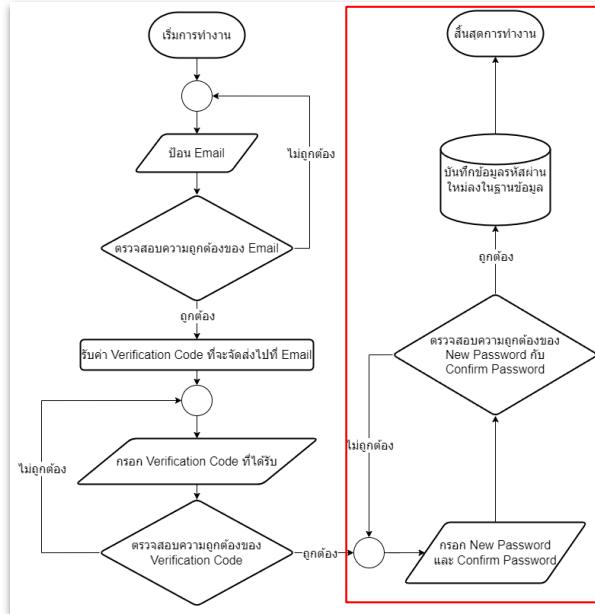
Forgot Password



```
// API forgot Code Verify  
// This function verifies a code entered by the user and redirects  
// otherwise it redirects them back to the forgot password page.  
var randomNUM int = int(sixDigits())  
  
func forgotCodeVerify(w http.ResponseWriter, r *http.Request) {  
    r.ParseForm()  
    fmt.Println(randomNUM)  
    codeSt := r.FormValue("forgotEmail")  
    codeInt, err := strconv.ParseInt(codeSt, 10, 64)  
    FetchError(err)  
    if randomNUM == int(codeInt) {  
        data := struct {  
            Code int  
        }{  
            Code: randomNUM,  
        }  
        err := updatePassView.Template.Execute(w, data)  
        FetchError(err)  
    } else {  
        http.Redirect(w, r, "/forgot_pass", http.StatusSeeOther)  
        fmt.Println("----- | No")  
    }  
}
```

Flow Chart

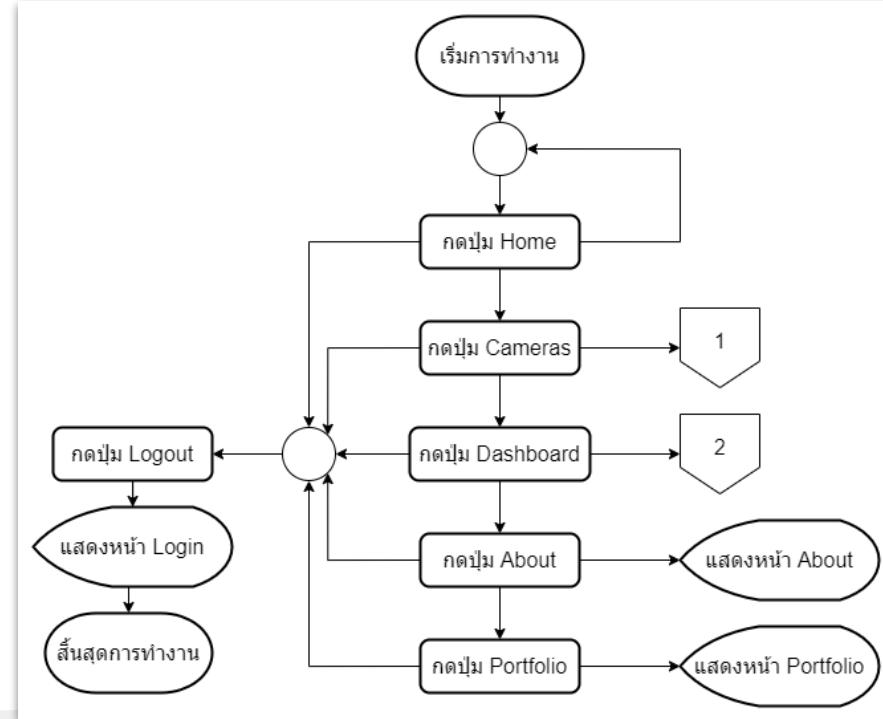
Forgot Password



```
// API Check Password  
// The function checks if two password inputs match and updates the password if they do, or returns an error if they don't.  
func checkPass(w http.ResponseWriter, r *http.Request) {  
    r.ParseForm()  
    pass1 := r.FormValue("pass1")  
    pass2 := r.FormValue("pass2")  
    if pass1 == pass2 {  
        UpdatePassword(pass1, isEmail)  
        http.Redirect(w, r, "/login", http.StatusSeeOther)  
    } else {  
        updatePassView.Template.Execute(w, "Please Make sure Your password Both of same")  
    }  
}  
  
// GetUpdatePassword  
// The function updates a user's password in the database.  
func UpdatePassword(upPass, useremail string) (bool, error) {  
    qs := fmt.Sprintf("UPDATE user SET Password = '%s' WHERE Email='%s'", upPass, useremail)  
    row := msq1.RawSQL(qs, db_user)  
    return row, nil  
}
```

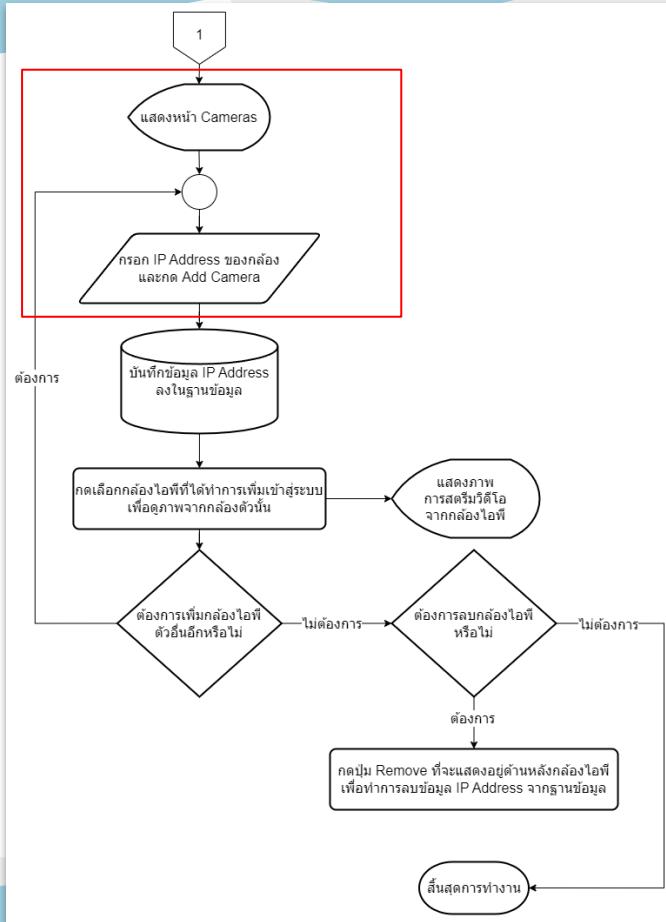
Flow Chart

After Login



Flow Chart

Cameras



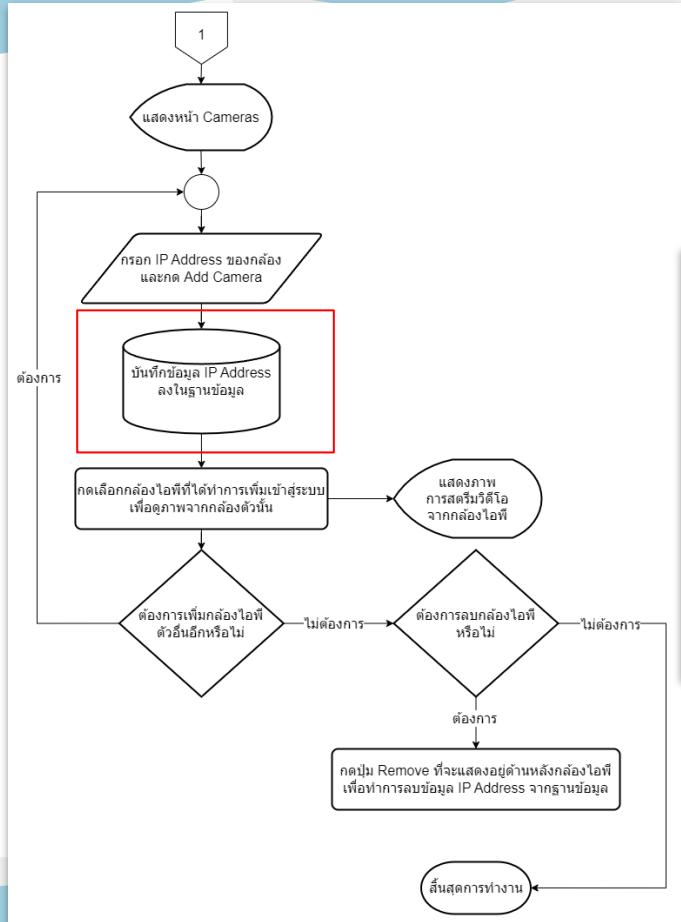
```
// API Cameras page
// This function retrieves camera information from the database
// If the user is not Logged in to the system will be redirected
func cameras(w http.ResponseWriter, r *http.Request) {
    session, err := store.Get(r, "login-session")
    FetchError(err)
    ok := session.Values["username"]
    if ok {
        rows, err := db_cameras.Query("SELECT url FROM cameras")
        if err != nil {
            log.Println(err)
            return
        }
        defer rows.Close()

        var cameraURLs []string
        for rows.Next() {
            var url sql.NullString
            err = rows.Scan(&url)
            if err != nil {
                log.Println(err)
                return
            }
            if url.Valid {
                cameraURLs = append(cameraURLs, url.String)
            } else {
                cameraURLs = append(cameraURLs, "")
            }
        }
        if err = rows.Err(); err != nil {
            log.Println(err)
            return
        }

        err = camerasView.Template.Execute(w, struct {
            CameraURLs []string
        }{
            CameraURLs: cameraURLs,
        })
        FetchError(err)
    } else {
        http.Redirect(w, r, "/login", http.StatusSeeOther)
    }
}
```

Flow Chart

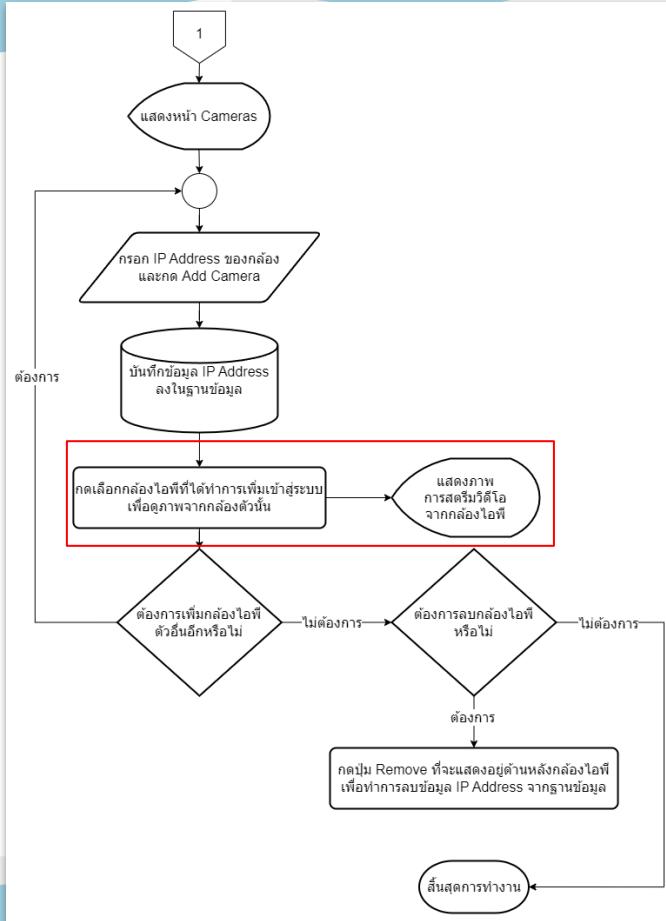
Cameras



```
// The function connects to a SQLite database for cameras and creates the cameras table if it does not exist.
func database_cameras() {
    // Connect to cameras database
    db_cameras, err = sql.Open("sqlite3", "./cameras.db")
    if err != nil {
        log.Fatal(err)
    }
    // Create cameras table if not exists
    createTableStmt := `CREATE TABLE IF NOT EXISTS cameras (
        id INTEGER PRIMARY KEY AUTOINCREMENT,
        url TEXT
    )`
    _, err = db_cameras.Exec(createTableStmt)
    if err != nil {
        log.Fatal(err)
    }
    db_cameras.SetMaxOpenConns(1)
    log.Println("| Database Cameras | Connection : Successful")
}
```

Flow Chart

Cameras



HTML

```
<div id="camera-stream" class="w3-padding"></div>

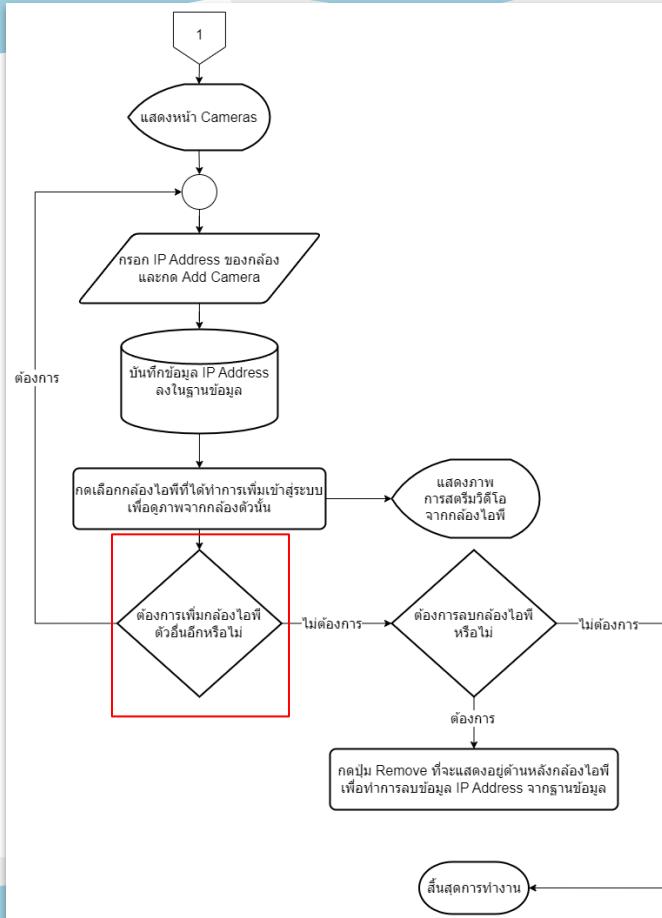
{{range .CameraURLs}}
<div class="w3-padding">
  <button class="w3-bar-item w3-blue w3-hover-pink w3-text-white" onclick="showCamera('{{.}}')">>{{.}}</button>
</div>
{{end}}
```

JavaScript

```
function showCamera(cameraURL) {
  var iframe = document.createElement("iframe");
  iframe.src = "http://" + cameraURL + "/video";
  iframe.className = "w3-card-4 w3-round-xlarge";
  iframe.width = "1600px";
  iframe.height = "1200px";
  iframe.frameBorder = "0";
  document.getElementById("camera-stream").innerHTML = "";
  document.getElementById("camera-stream").appendChild(iframe);
}
```

Flow Chart

Cameras



API

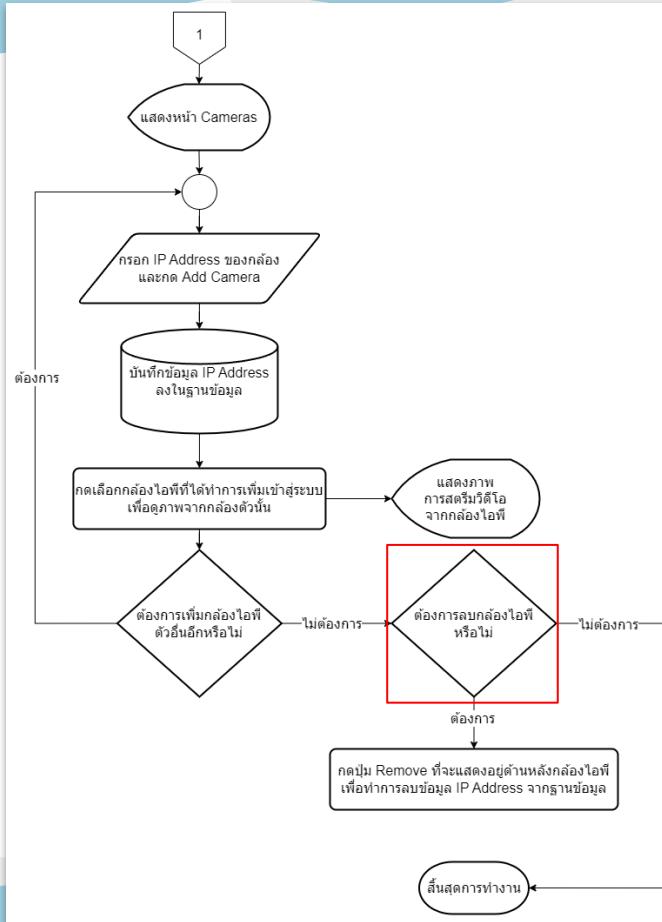
```
// API Add Camera
// The function adds a camera URL to a database table and returns a response
func addCamera(w http.ResponseWriter, r *http.Request) {
    cameraURL := r.FormValue("cameraURL")
    insertStmt := `INSERT INTO cameras (url) VALUES (?)`
    _, err := db_cameras.Exec(insertStmt, cameraURL)
    if err != nil {
        log.Println(err)
        return
    }
    w.WriteHeader(http.StatusOK) // sends a reply that the operation was successful
}
```

JavaScript

```
function addCamera() {
    var cameraURL = document.getElementById("camera-url-input").value;
    if (cameraURL) {
        var cameraList = document.getElementById("camera-stream");
        var button = document.createElement("button");
        button.className = "w3-bar-item w3-button w3-blue w3-hover-pink";
        button.innerHTML = cameraURL;
        button.onclick = function () {
            showCamera(cameraURL);
        };
        cameraList.appendChild(button);
        document.getElementById("camera-url-input").value = "";
    }
    // Send data to the Add Camera API.
    fetch("/add_camera", {
        method: "POST",
        body: new URLSearchParams({
            cameraURL: cameraURL
        })
    });
    // refresh the page
    location.reload();
}
```

Flow Chart

Cameras



API

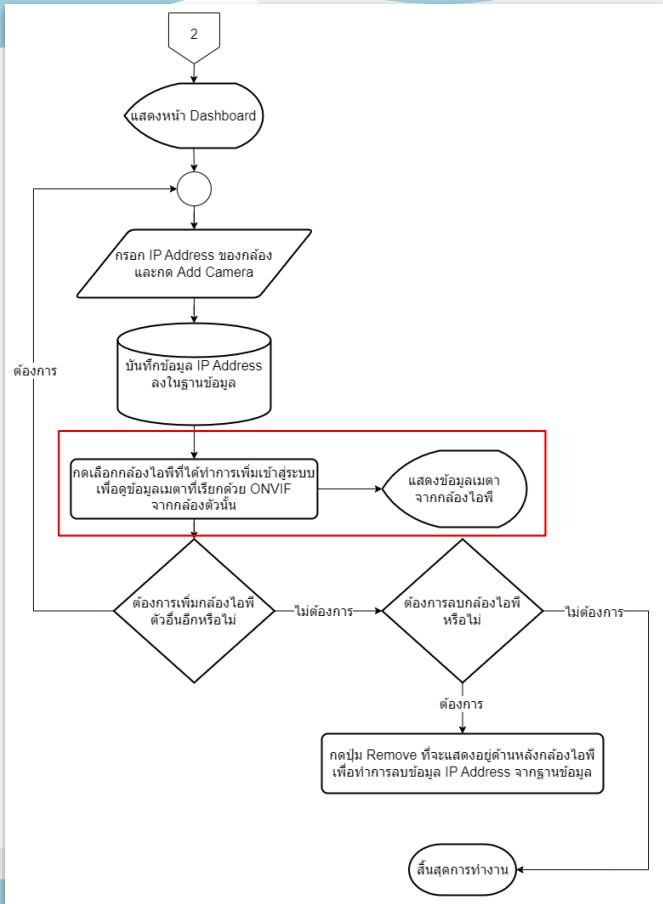
```
// API Remove Camera
// This function removes camera data from a database based on its URL
func removeCamera(w http.ResponseWriter, r *http.Request) {
    cameraURL := r.FormValue("cameraURL")
    deleteStmt := `DELETE FROM cameras WHERE url = ?` // delete statement
    _, err := db_cameras.Exec(deleteStmt, cameraURL)
    if err != nil {
        log.Println(err)
        return
    }
    w.WriteHeader(http.StatusOK) // sends a reply that the operation was successful
}
```

JavaScript

```
function removeCamera(cameraURL) {
    // Send data to the Remove Camera API.
    fetch("/remove_camera", {
        method: "POST",
        body: new URLSearchParams({
            cameraURL: cameraURL
        })
    }).then(function (response) {
        if (response.ok) {
            // camera remove button
            var cameralist = document.getElementById("camera-stream");
            var buttons = cameralist.getElementsByTagName("button");
            for (var i = 0; i < buttons.length; i++) {
                if (buttons[i].innerHTML === cameraURL) {
                    buttons[i].parentNode.removeChild(buttons[i]);
                    break;
                }
            }
        }
    });
    // refresh the page
    location.reload();
}
```

Flow Chart

Dashboard



```
// API metadatas
// The function retrieves metadata information from an ONVIF camera
func metadatas(w http.ResponseWriter, r *http.Request) {
    cameraURL := r.FormValue("cameraURL")
    Camera := onvif.NewOnvifDevice(cameraURL)
    camera.Auth("admin", "admin") // "user", "password"

    err := camera.Initialize()
    if err != nil {
        http.Error(w, err.Error(), http.StatusInternalServerError)
        return
    }
    di, err := camera.Device.GetDeviceInformation()
    if err != nil {
        http.Error(w, err.Error(), http.StatusInternalServerError)
        return
    }
    users, err := camera.Device.GetUsers()
    if err != nil {
        http.Error(w, err.Error(), http.StatusInternalServerError)
        return
    }
    dateandtime, err := camera.Device.GetSystemDateAndTime()
    if err != nil {
        http.Error(w, err.Error(), http.StatusInternalServerError)
        return
    }
    cap, err := camera.Device.GetCapabilities()
    if err != nil {
        http.Error(w, err.Error(), http.StatusInternalServerError)
        return
    }
    hostname, err := camera.Device.GetHostname()
    if err != nil {
        http.Error(w, err.Error(), http.StatusInternalServerError)
        return
    }
    networkprotocols, err := camera.Device.GetNetworkProtocols()
    if err != nil {
        http.Error(w, err.Error(), http.StatusInternalServerError)
        return
    }
    discovery, err := camera.Device.GetDiscoveryMode()
    if err != nil {
        http.Error(w, err.Error(), http.StatusInternalServerError)
        return
    }
    servicecap, err := camera.Device.GetServiceCapabilities()
    if err != nil {
        http.Error(w, err.Error(), http.StatusInternalServerError)
        return
    }
}
```

ONVIF 2.0 Service

Function



DeviceMgmt

GetDeviceInformation	SetSystemDateAndTime	GetSystemDateAndTime	SetSystemFactoryDefault
UpgradeSystemFirmware	SystemReboot	RestoreSystem	GetSystemBackup
GetSystemLog	GetSystemSupportInformation	GetScopes	SetScopes
AddScopes	RemoveScopes	GetDiscoveryMode	SetDiscoveryMode
GetRemoteDiscoveryMode	SetRemoteDiscoveryMode	GetIPAddresses	SetIPAddresses
GetEndpointReference	GetRemoteUser	SetRemoteUser	GetUsers
CreateUsers	GetReliableUsers	SetUser	GetVirtualPort
GetCapabilities	GetHostname	SetHostname	GetDNS
SetDNS	GetNTP	SetNTP	GetDynamicDNS
SetDynamicDNS	GetNetworkInterfaces	SetNetworkInterfaces	GetNetworkProtocols
SetNetworkProtocols	GetNetworkDefaultGateway	SetNetworkDefaultGateway	GetZeroConfiguration
SetZeroConfiguration	GetIPAddressFilter	SetIPAddressFilter	AddIPAddressFilter
RemoveIPAddressFilter	GetAccessPolicy	SetAccessPolicy	CreateCertificate
GetCertificates	GetCertificatesStatus	SetCertificatesStatus	DeleteCertificates
GetPkcs10Request	LoadCertificates	GetClientCertificateMode	SetClientCertificateMode
GetRelayOutputs	SetRelayOutputSettings	SetRelayOutputState	SendAuxiliaryCommand
GetCACertificates	LoadCertificateWithPrivateKey	GetCertificateInformation	LoadCACertificates
CreateDot1XConfiguration	SetDot1XConfiguration	GetDot1XConfiguration	GetDot1XConfigurations
DeleteDot1XConfiguration	GetDot11Capabilities	GetDot11Status	ScanAvailableDot11Networks
GetSystemUris	StartFirmwareUpgrade	StartSystemRestore	

DeviceIO

GetAudioSources	GetAudioOutputs	GetVideoSources	GetVideoOutputs
GetVideoSourceConfiguration	GetVideoOutputConfiguration	Get AudioSource Configuration	Get AudioOutput Configuration
SetVideoSourceConfiguration	SetVideoOutputConfiguration	Set AudioSource Configuration	Set AudioOutput Configuration
GetVideoSourceConfigurationOptions	GetVideoOutputConfigurationOptions	Get AudioSource Configuration Options	Get AudioOutput Configuration Options
GetRelayOutputs	SetRelayOutputSettings	SetRelayOutputState	

Event

CreatePullPointSubscription	GetEventProperties	PullMessages	SetSynchronizationPoint
-----------------------------	--------------------	--------------	-------------------------

Analytics

GetSupportedRules	CreateRules	DeleteRules	GetRules
ModifyRules	GetSupportedAnalyticsModules	CreateAnalyticsModules	DeleteAnalyticsModules
GetAnalyticsModules	ModifyAnalyticsModules		

Operation

5.1. GetSystemDateAndTime

Description:

This operation gets the device system date and time. The device shall support the return of the daylight saving setting and of the manual system date and time (if applicable) or indication of NTP time (if applicable) through the GetSystemDateAndTime command. A device shall provide the UTCDateTime information.

SOAP action:

<http://www.onvif.org/ver10/device/wsdl/GetSystemDateAndTime>

Input:

[GetSystemDateAndTime]

Output:

[GetSystemDateAndTimeResponse]

▪ **SystemDateAndTime** [SystemDateTime]
Contains information whether system date and time are set manually or by NTP, daylight savings is on or off, time zone in POSIX 1003.1 format and system date and time in UTC and also local system date and time.

▪ **DateType** [TypeDateTime]

Indicates if the time is set manually or through NTP.

- enum { 'Manual', 'NTP' }

▪ **DaylightSavings** [Boolean]

Informational indicator whether daylight savings is currently on/off.

▪ **TimeZone** [optional]; [TimeZone]

Timezone information in Posix format.

▪ **TZ** [token]

Posix timezone string.

▪ **UTCDateTime** - optional; [DateTime]

Current system date and time in UTC format. This field is mandatory since version 2.0.

▪ **Time** [Time]

▪ **Hour** [int]

Range is 0 to 23.

▪ **Minute** [int]

Range is 0 to 59.

▪ **Second** [int]

Range is 0 to 61 (typically 59).

▪ **Date** [Date]

▪ **Year** [int]

Range is 1 to 9999.

▪ **Month** [int]

Range is 1 to 12.

▪ **Day** [int]

Range is 1 to 31.

▪ **LocalDateTime** [optional]; [DateTime]

Date and time in local format.

▪ **Time** [Time]

▪ **Hour** [int]

Range is 0 to 23.

▪ **Minute** [int]

Range is 0 to 59.

▪ **Second** [int]

Range is 0 to 61 (typically 59).

▪ **Date** [Date]

▪ **Year** [int]

Range is 1 to 9999.

▪ **Month** [int]

Range is 1 to 12.

▪ **Day** [int]

Range is 1 to 31.

▪ **Extension** - optional; [SystemDateTimeExtension]

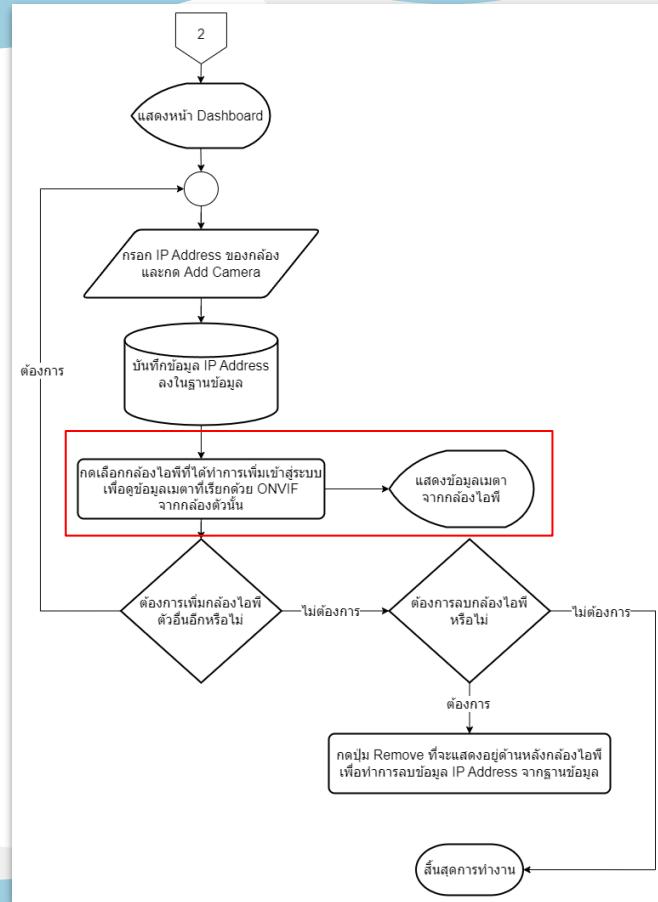
ONVIF Data

ข้อมูลที่เรียกได้เบื้องต้น

```
PROBLEMS OUTPUT GITLENS TERMINAL
9:36:25 main | -----
9:36:25 main | Waiting (loop 3)...
9:36:25 app | 2023/06/29 09:36:25 | Database User | Connection : Successful
9:36:25 app |
9:36:25 app | 2023/06/29 09:36:25 | Database Cameras | Connection : Successful
9:36:25 app | Listening port : 9000
9:37:45 app | Data From GetDeviceInformation : {Pavel Khlebovich IP Webcam 849 SN000000 Android Device}
9:37:45 app | Data From GetUsers : {[{admin admin Administrator <nil>}]}}
9:37:45 app | Data From GetSystemDateAndTime : {{Manual false {} 0xc0002b4420 {{2 37 44} {2023 5 29}} {{} [] [] }}}
9:37:45 app | Data From GetCapabilities : {{<nil> 0xc0000c6fc0 <nil> 0xc0001c2230 0xc0000f8880 <nil> <nil>}}
9:37:45 app | Data From GetHostname : {{false IP Webcam @ 10.20.3.10:8080 <nil>}}
9:37:45 app | Data From GetNetworkProtocols : {[{RTSP true 8080 <nil>}]}
9:37:45 app | Data From GetDiscoveryMode : {Discoverable}
9:37:45 app | Data From GetServiceCapabilities : {{{{false false false false false 0 false 0 false} {false false false false false se false false 0 0 } <nil>}}}
```

Flow Chart

Dashboard



```
// Generate JSON data from di, users, dateandtime, cap, data := struct {  
    DeviceInformation interface{}  
    Users interface{}  
    SystemDateAndTime interface{}  
    Capabilities interface{}  
    Hostname interface{}  
    NetworkProtocols interface{}  
    DiscoveryMode interface{}  
    ServiceCapabilities interface{}  
}{  
    DeviceInformation: di,  
    Users: users,  
    SystemDateAndTime: dateandtime,  
    Capabilities: cap,  
    Hostname: hostname,  
    NetworkProtocols: networkprotocols,  
    DiscoveryMode: discovery,  
    ServiceCapabilities: servicecap,  
}  
  
// Sends data back to the web page in JSON response  
w.Header().Set("Content-Type", "application/json")  
json.NewEncoder(w).Encode(data)  
}
```

Table Showing all APIs

ลำดับ	API		คำอธิบาย
การเข้าสู่ระบบ			
1	GET	/	แสดงหน้าหลักในระบบ
2	GET	/cameras	แสดงหน้าจัดการกล้องไอโอพี
3	GET	/dashboard	แสดงหน้าแสดงข้อมูลเมตาจากกล้องไอโอพี
4	GET	/about	แสดงหน้าข้อมูลตารางการทำงานของโครงการ
5	GET	/portfolio	แสดงหน้าข้อมูลของนักพัฒนา
6	GET	/login	แสดงหน้าการเข้าสู่ระบบ
7	POST	/loginauth	การใส่ข้อมูล email และ password เพื่อเข้าระบบ
8	GET	/signup	แสดงหน้าการลงทะเบียนสร้างบัญชีผู้ใช้
9	POST	/signupauth	การใส่ข้อมูลเพื่อลงทะเบียนสร้างบัญชีผู้ใช้
10	GET	/forgot_pass	แสดงหน้าลืมรหัสผ่าน
11	POST	/code_verify	ตรวจสอบรหัสยืนยัน เพื่อเปลี่ยนรหัสผ่านใหม่
12	POST	/forgot_pass_auth	ตรวจสอบอีเมลที่ใช้ในการส่งรหัสยืนยัน
13	POST	/checkpass	สร้างและตรวจสอบรหัสผ่านใหม่
14	-	/ “เรียก API ที่ไม่รู้อยู่”	เมื่อเขียนเรียก API ที่ไม่ได้มีการสร้างไว้ จะแสดงหน้า 404 Not Fount แจ้งเตือน
การจัดการกล้องไอโอพี			
1	POST	/add_camera	การเพิ่มกล้องไอโอพี
2	POST	/remove_camera	การลบกล้องไอโอพี
การแสดงข้อมูลเมตาจากกล้องไอโอพี			
1	POST	/metedatas	การแสดงข้อมูลเมตาจากกล้องไอโอพี

04

Conclude

อธิบายถึงผลการดำเนินการและผลการทดลอง

การรวมของระบบ NONVIF

Login

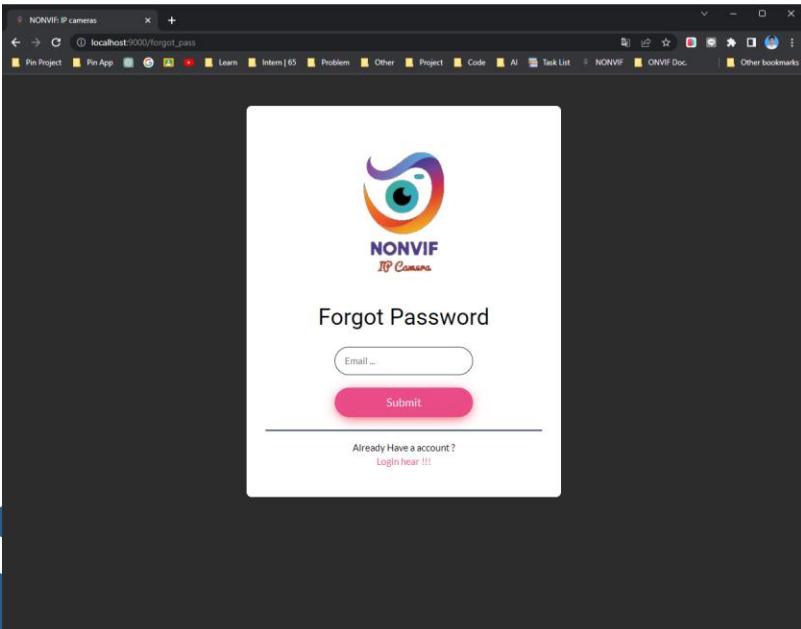
The screenshot shows a login form titled "Login". At the top is the NONVIF logo, which features a stylized eye icon with orange, yellow, and blue hues. Below the logo, the text "NONVIF" and "IP Camera" is displayed. The main form has two input fields: "Email ..." and "Password ...". Below these fields is a red link "Forgot password !!!". A large pink "Login" button is centered at the bottom. At the very bottom of the form, there are two small links: "Don't have an account ?" and "Create Your Account". The background of the page is white, and it is displayed in a browser window titled "NONVIF IP cameras" with the URL "localhost:3000/login".

Sign up

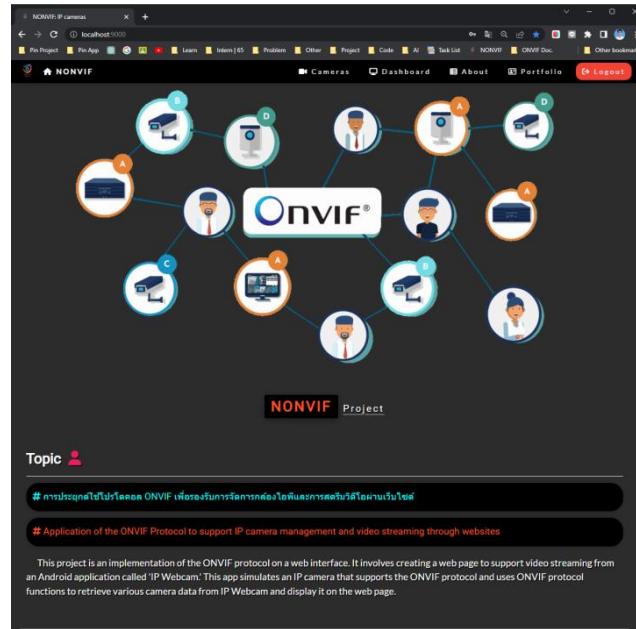
The screenshot shows a sign-up form titled "Sign up". It features the same stylized NONVIF logo at the top. The form contains five input fields: "Enter your Full Name", "Enter Your Email", "Enter Your User Name", "Enter Your Mobile Number", and "Enter Your Password". A large pink "Sign Up" button is located at the bottom. At the bottom of the form, there is a red link "Already Have a account ?" and a blue link "Login here". The background is white, and it is shown in a browser window titled "NONVIF IP cameras" with the URL "localhost:3000/signup".

การรวมของระบบ NONVIF

Forgot Password

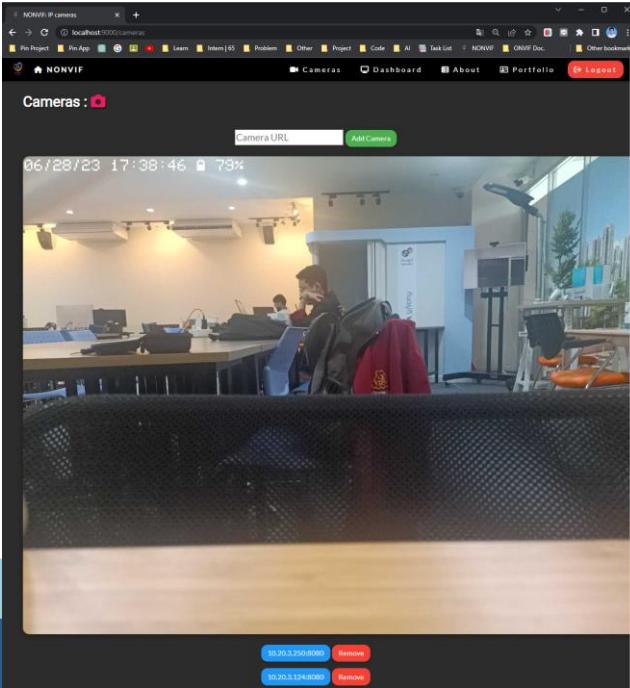


Home



การรวมของระบบ NONVIF

Cameras



Dashboard

The screenshot shows the NONVIF dashboard with the following configuration details:

- Hostname:** Hostname : IP Webcam @ 10.20.3.250:8080
FromDHCP : false
- NetworkProtocols:** Name : RTSP
Enabled : true
- Device Information:** Manufacturer : Pavel Khlebovich
Model : IP Webcam
Firmware Version : 849
Serial Number : SNO00000
Hardware ID : Android Device
- Users:** Username : admin
Password : admin
UserLevel : Administrator

การทดสอบ



Install

การติดตั้ง
NONVIF



Systems

Login, Logout,
Sign up, Forgot Password



Stream Video

การเรียกสตรีมภาพจาก
กล้องไอพี



Metadata

การเรียกข้อมูลเมตาจาก
กล้องไอพี



Other

การทดสอบอื่น ๆ

Install

The screenshot shows the GitHub repository page for `Nuntakorn-Buu / NONVIF`. The repository has 3 branches and 0 tags. The README.md file contains the following text:

```
ค่าใช้จ่ายในการซื้อขายของ NONVIF ที่มีอยู่ใน  
การค้าปลีกและห้างสรรพสินค้าทั่วไป  
ที่ไม่รู้ / Application of the ONVIF  
Protocol to support IP camera  
management and video streaming  
through websites .
```

The repository structure includes files like `tmp`, `vendor`, `view`, `gitignore`, `README.md`, `cameras.db`, `database.go`, `dbquery.go`, `email.go`, `go.mod`, `go.sum`, `golang_loginSystem`, `lib.go`, `loginSystem.db`, `loginSystem.db.sqlpro`, `main.go`, and `router.go`.

The `README.md` file also contains sections for **Packages** (No packages published) and **Languages** (HTML 68.5%, Go 27.3%, CSS 4.2%).

Suggested Workflows include `Actions Importer` and `SLA Go releaser`.

The screenshot shows the GitHub repository page for `Nuntakorn-Buu / NONVIF`. The `README.md` file contains the following content:

Component
You must have these installed first, to install the NONVIF project.

Software

- Ubuntu 20.04.3 LTS
 - Installation method -> [Link](#)
- Coling Version 1.19
 - Installation method -> [Link](#)

Hardware

- At least 2 IP cameras
 - or
 - use the "IP Webcam" application installed on your Android phone | version 1.166.783
 - Installation method -> [Link](#)

Dev Tool

- VSCode or Visual Studio Code version 1.76
 - Installation method -> [Link](#)

Installation
Install the `NONVIF` project with the following command.

Step : 1
Please clone this repository from Nuntakorn-Buu/NONVIF on Github.
`git clone https://github.com/nuntakorn-buu/nonvif.git`

Step : 2
After completing the repository clone, use the following command to navigate into the folder NONVIF
`cd NONVIF`

Step : 3
After that, use the command `go get .` to install and update the necessary modules for this project.
`go get .`

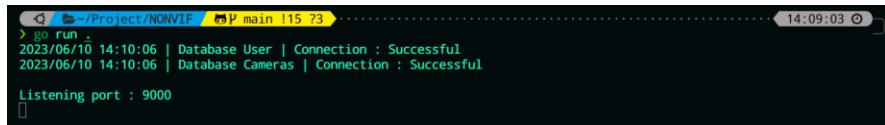
Running Tests
To run tests, Run the following command
`go run .`

If the `go run .` command is successful, you will see the following message below.

```
2023/06/10 14:10:00 | Database User | Connection : Successful
2023/06/10 14:10:00 | Database Schema | Connection : Successful
Listening port : 9000
```

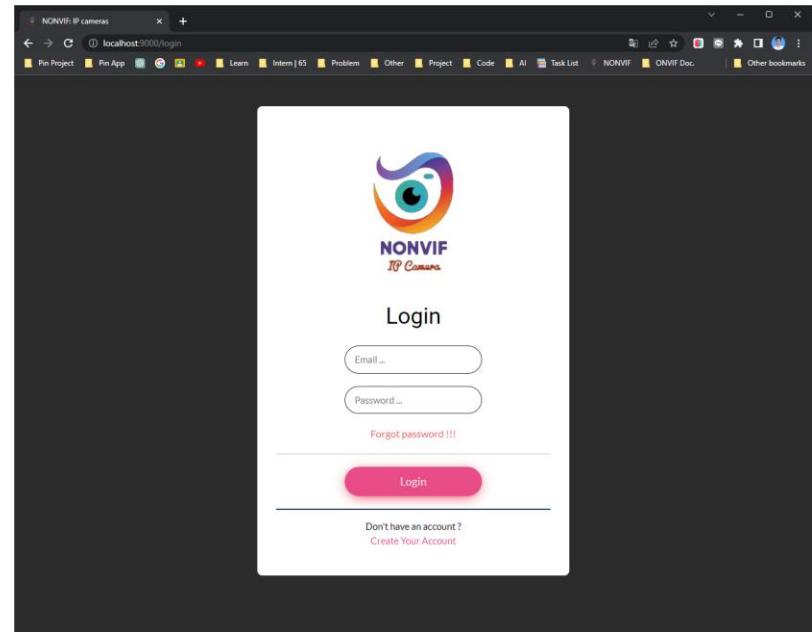
Install

go run .



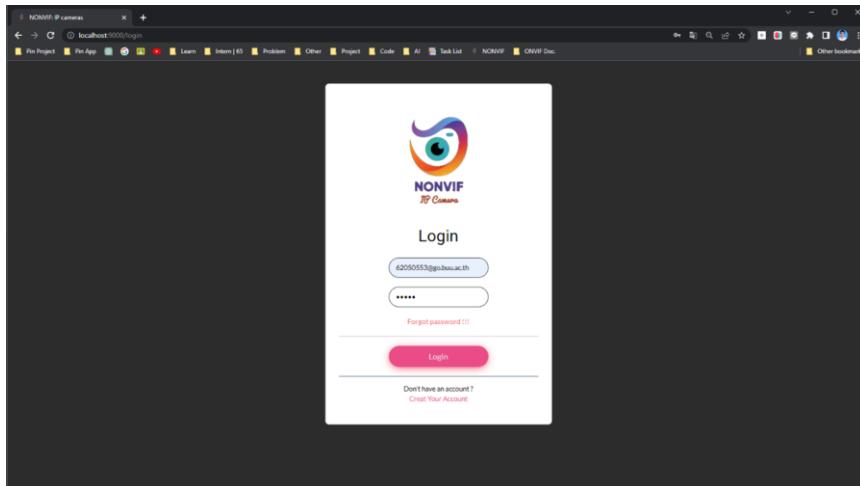
```
14:09:03
> go run .
2023/06/10 14:10:06 | Database User | Connection : Successful
2023/06/10 14:10:06 | Database Cameras | Connection : Successful
Listening port : 9000
```

localhost:9000

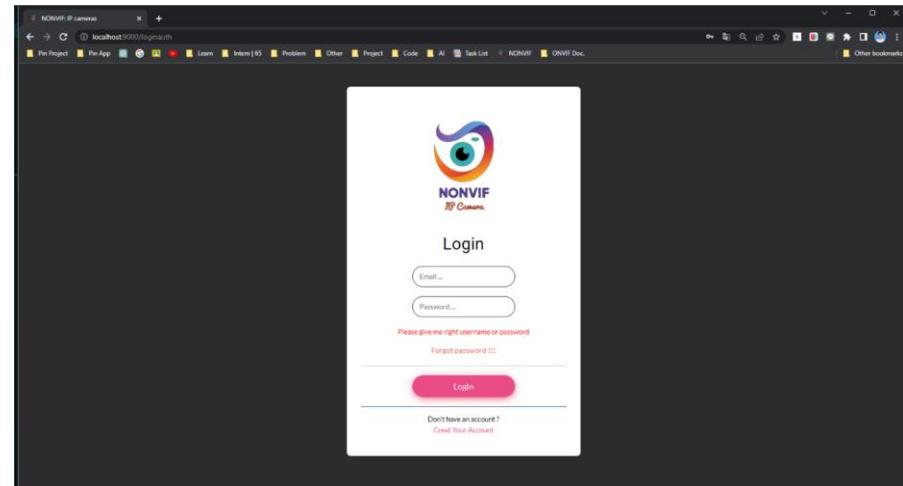


Systems

Login เข้าระบบ



Login ผิดพลาด



Systems

Sign up ลงทะเบียน

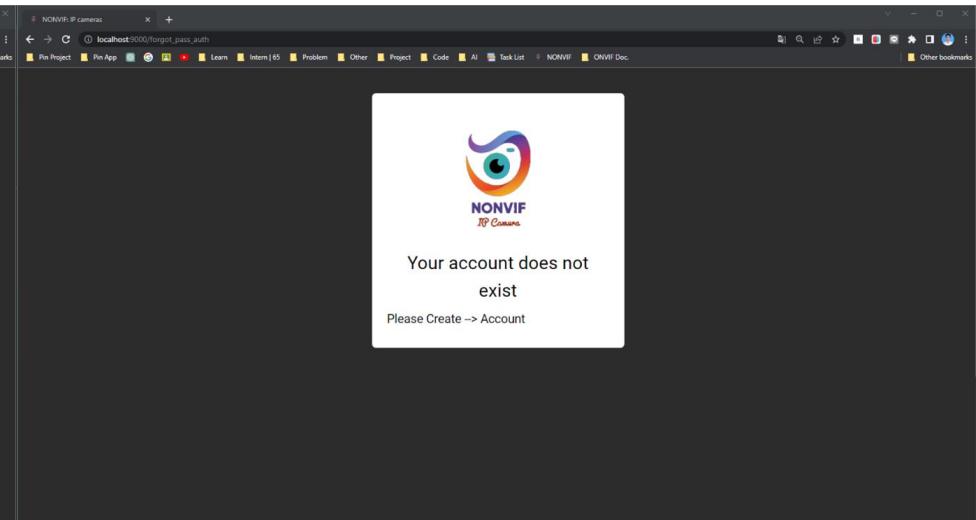
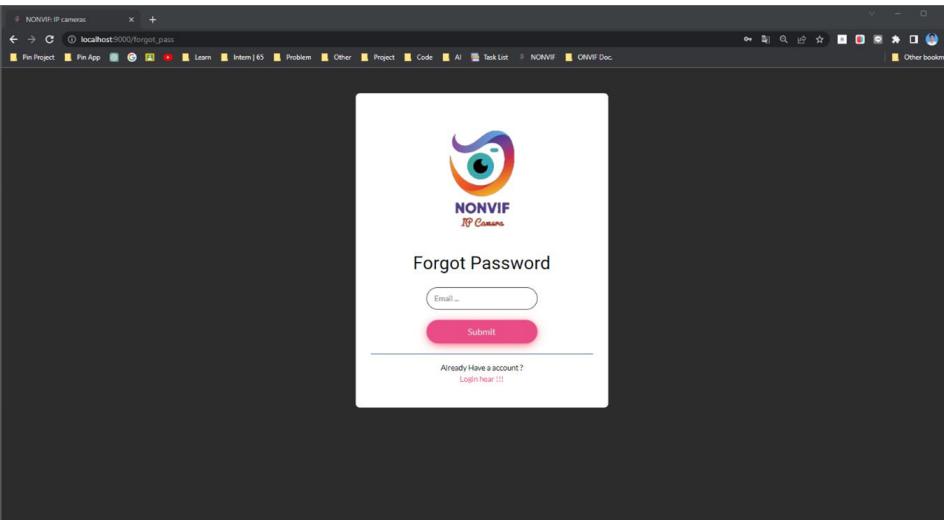
Sign up ผิดพลาด

The screenshot shows a sign-up form for 'NONVIF IP Cameras'. The form includes fields for 'Enter your Full Name', 'Enter Your Email', 'Enter Your User Name', 'Enter Your Mobile Number', and 'Enter Your Password'. A red 'Sign Up' button is at the bottom. Below the form is a link 'Already Have a account? [Login/Here](#)'.

The screenshot shows a sign-up form for 'NONVIF IP Cameras'. The form includes fields for 'Enter your Full Name' and 'Enter Your Email'. An error message box appears, stating 'Please include an "@" in the email address. 42342 is missing an "@"'. Below the form is a link 'Already Have a account? [Login/Here](#)'.

Systems

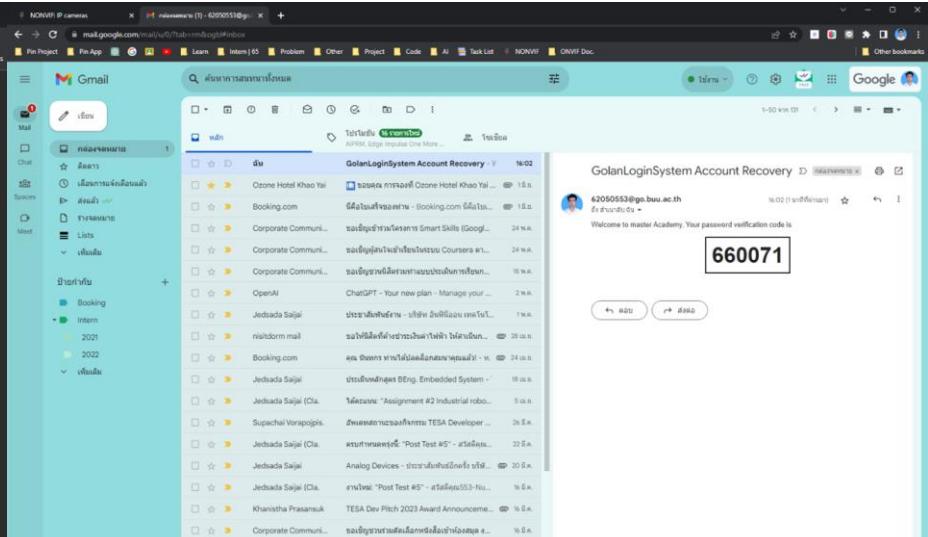
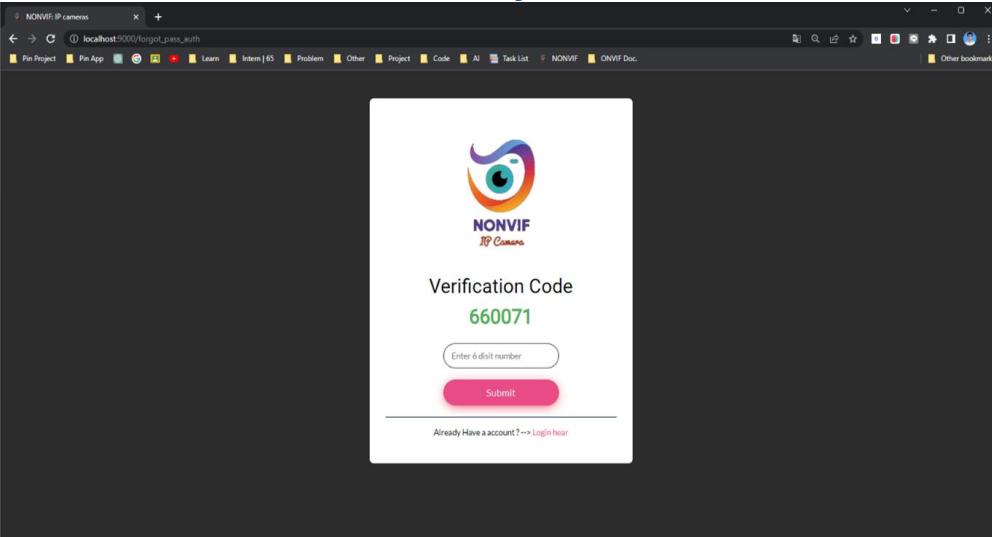
Forgot Password



Forgot Password Email ผิดพลาด

Systems

Forgot Password
Email ถูกต้อง



Systems

กรอก Verification
Code ที่ได้รับ

NONVIF IP cameras

localhost:9000/forgot_pass_auth

Verification Code
660071

660071

Submit

Already Have a account ? >> [Login here](#)

สร้างรหัสผ่านใหม่

NONVIF IP cameras

localhost:9000/code.verify

New Password

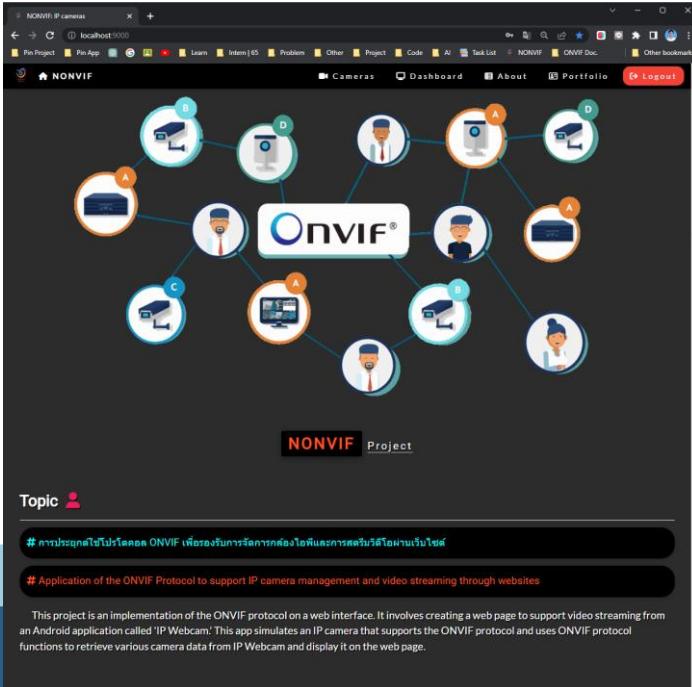
Enter your new password

Confirm your new password

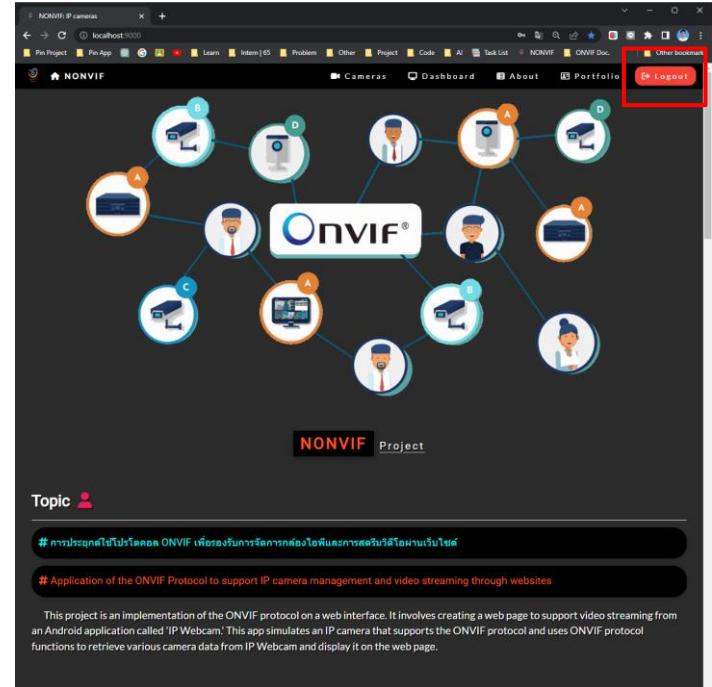
Submit

Systems

เข้าสู่หน้า Home
ภายในระบบ



การ Logout

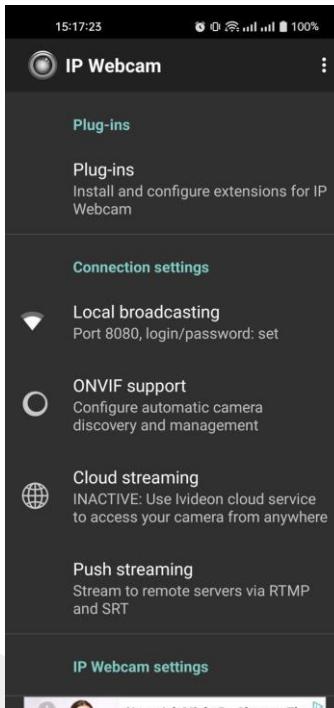


Stream Video

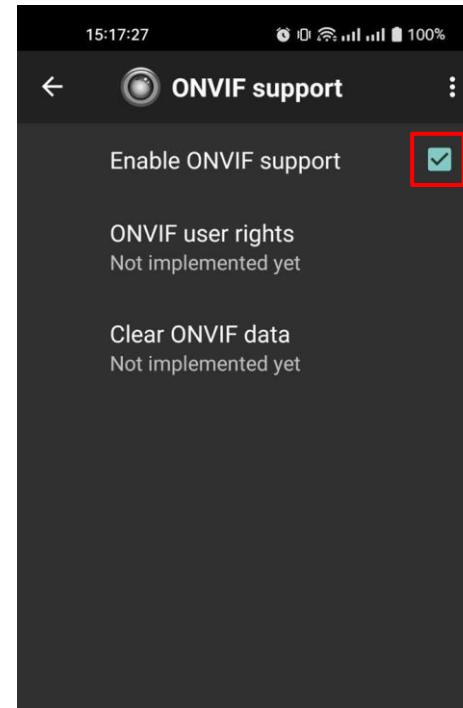
เข้าหน้า Download



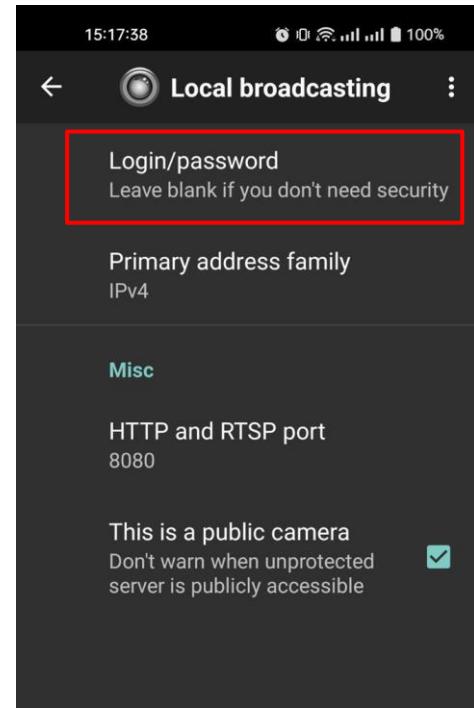
เข้าสู่ภายในแอพ
IP Webcam



เข้าหน้า Download

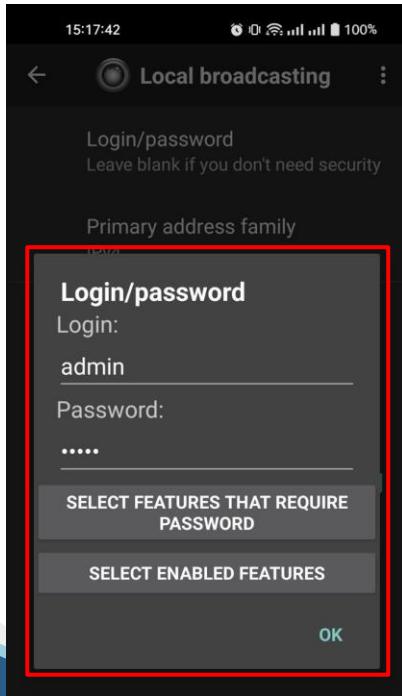


เข้าสู่ภายในแอพ
IP Webcam

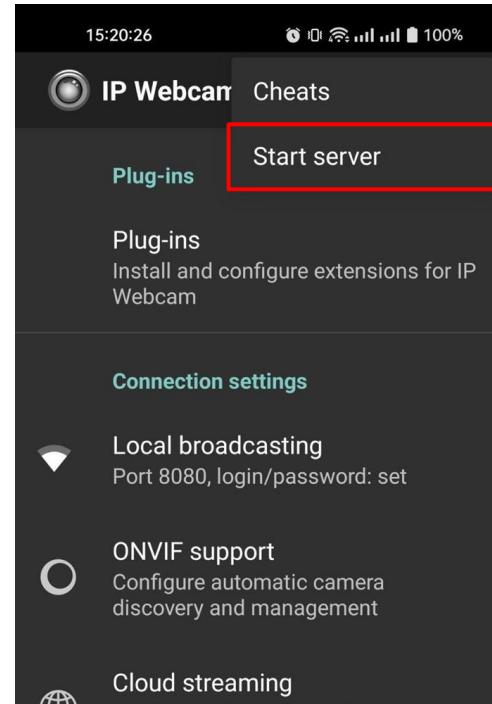


Stream Video

กรอกรหัสผ่าน
สำหรับกล้อง



เริ่มต้นการจำลอง
กล้องไอโอพี

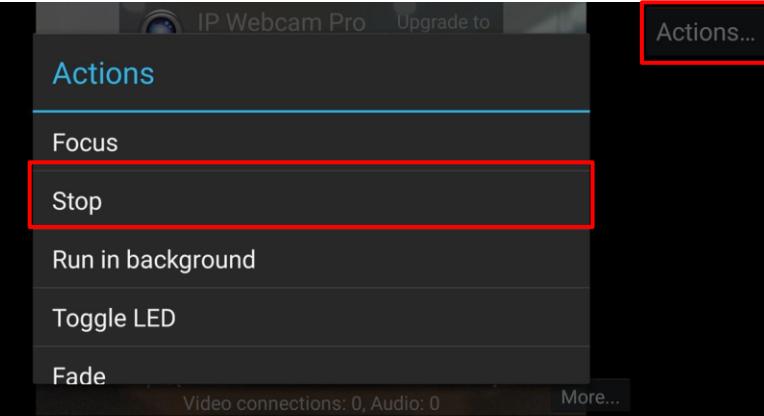


Stream Video

กรอกรหัสผ่าน
สำหรับกล้อง

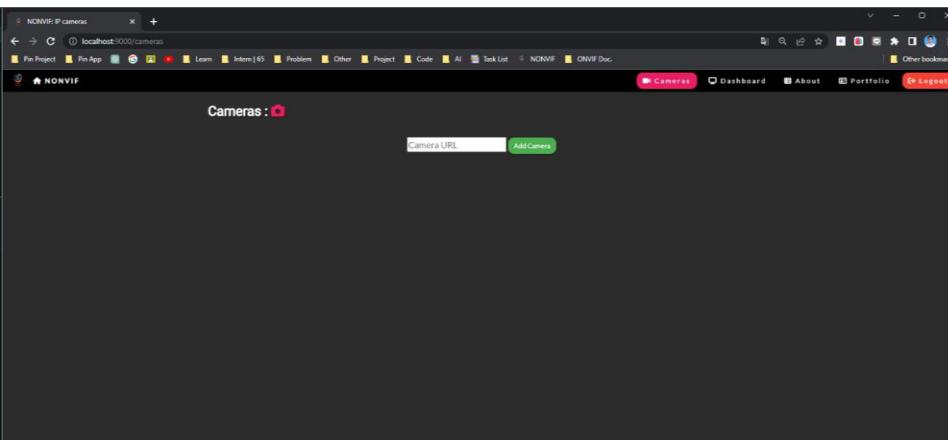


หยุดการทำงาน
การจำลองกล้อง

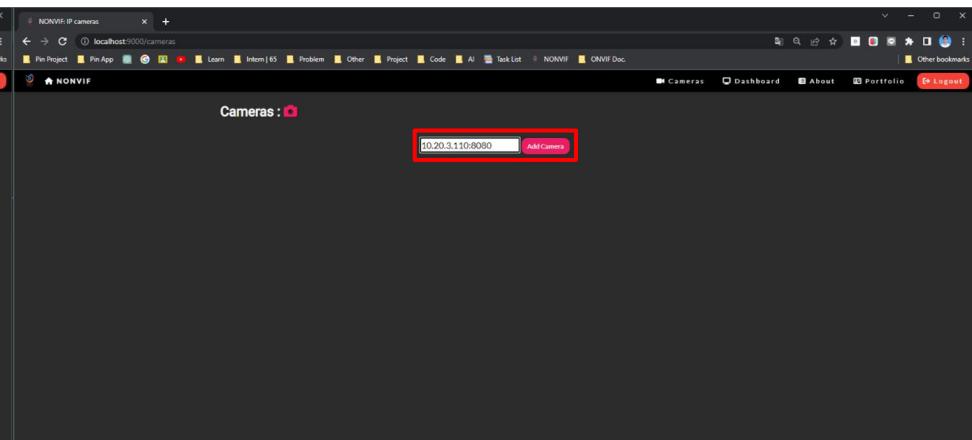


Stream Video

หน้า Cameras

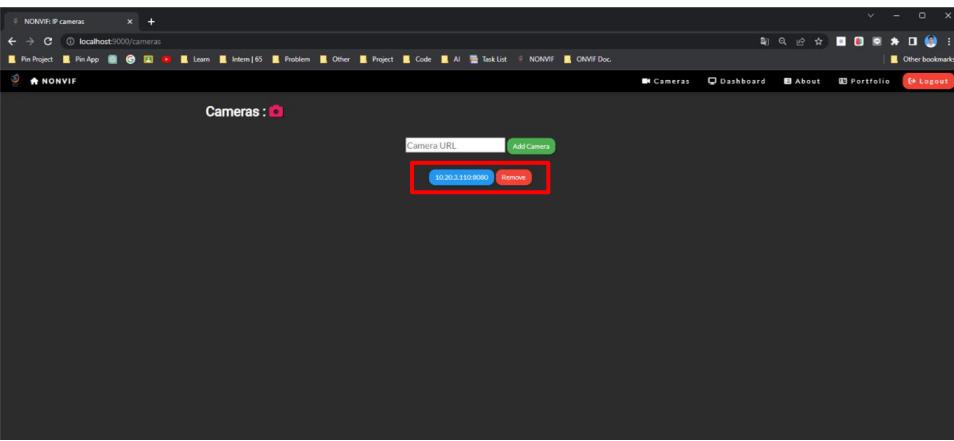


นำ IP Address
มากรอก

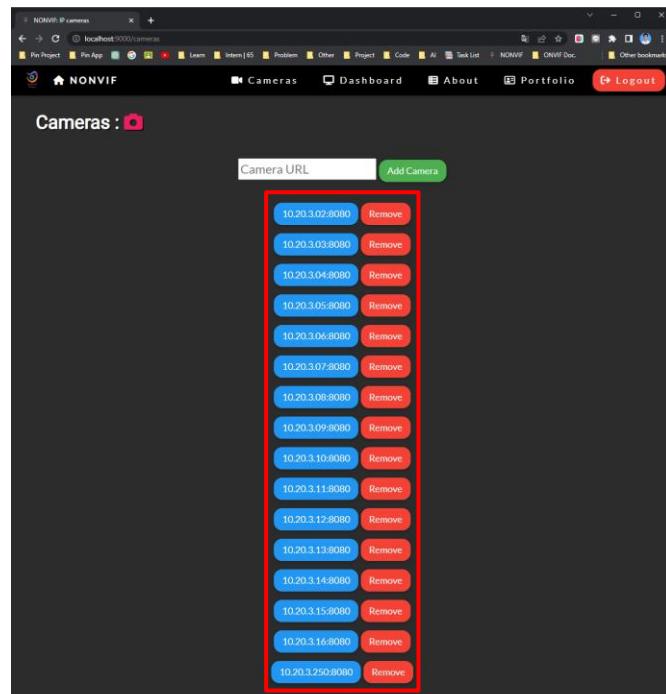


Stream Video

แสดง IP Address
ของกล้องที่ถูกเพิ่ม

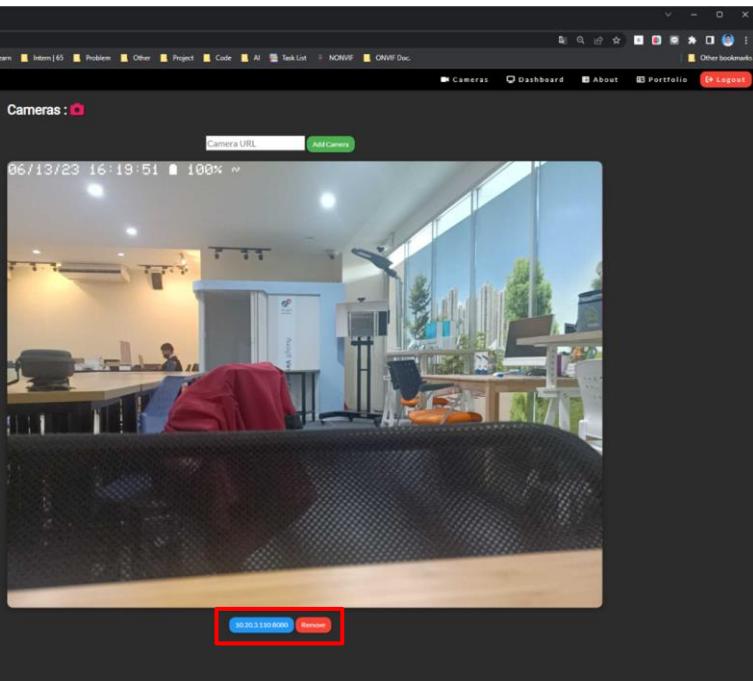


แสดงรายการกล้อง
ที่ถูกกรอกเพิ่ม

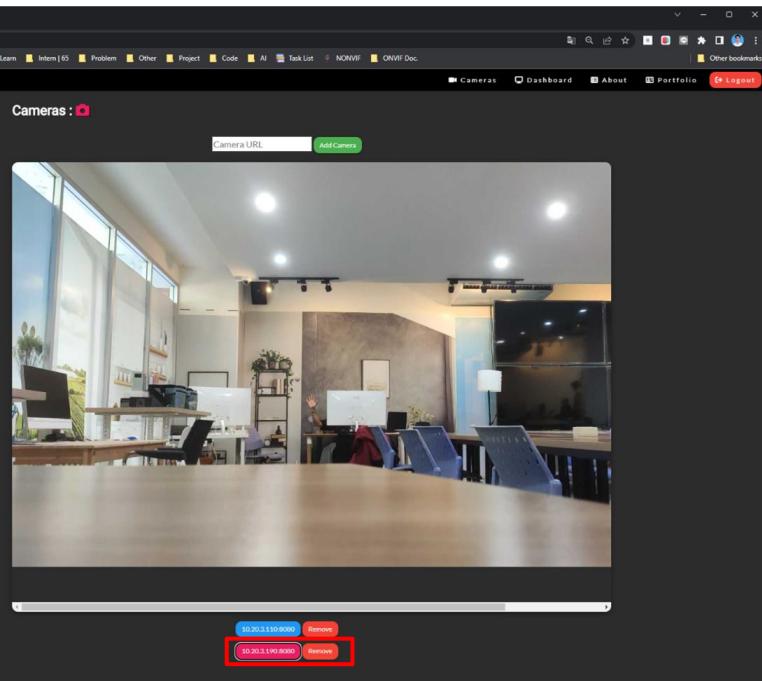


Stream Video

การสตรีมวิดีโอ¹
จากกล้องไอพี

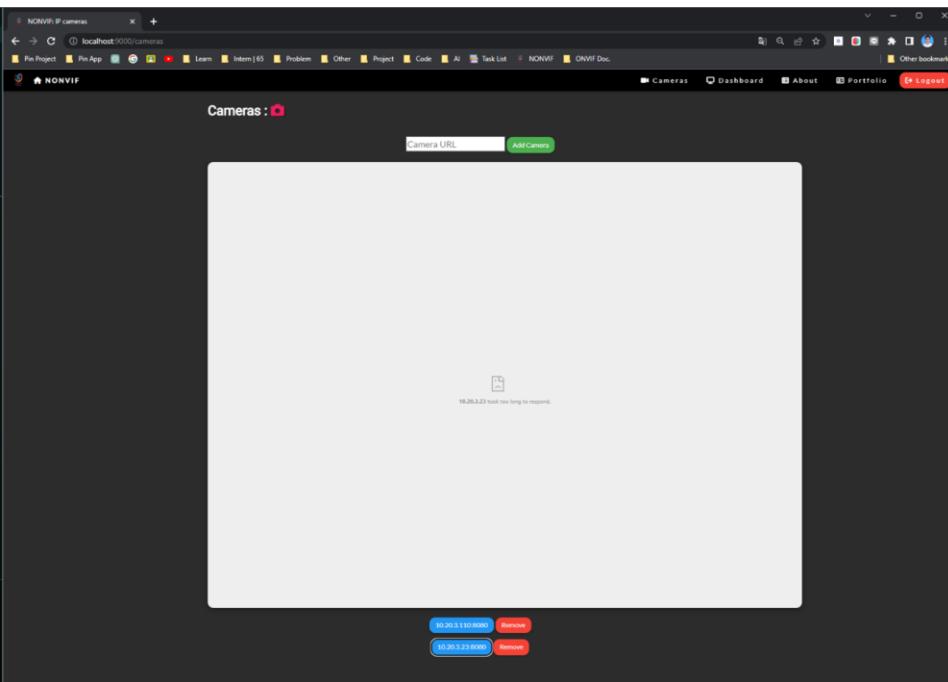


การสตรีมวิดีโอ²
จากกล้องไอพี

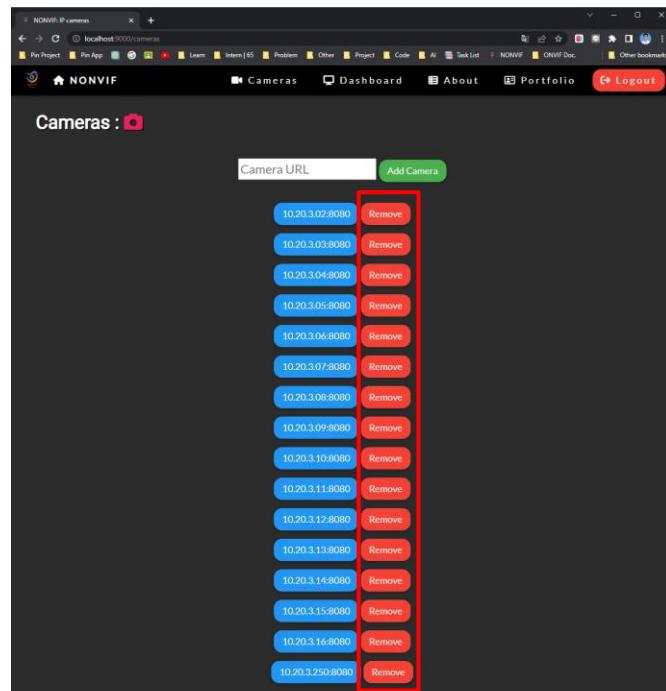


Stream Video

กล้องที่เพิ่มเข้ามา
แบบผิดพลาด



การลบ IP Address
กล้องไอพี



Metadata

GetHostname,
GetNetwork,
GetDeviceInformation

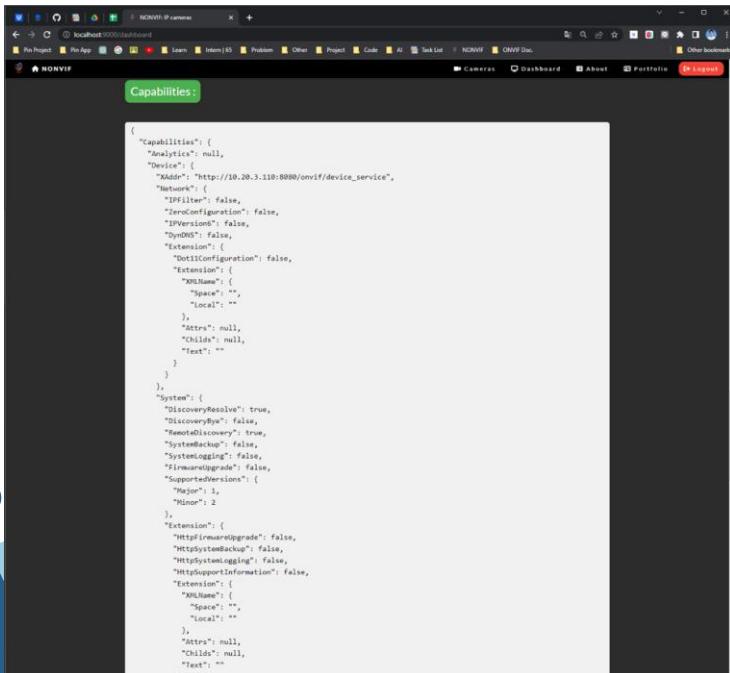
```
Dashboard : ?  
  
Hostname :  
{  
    "HostnameInformation": {  
        "FromDHCP": false,  
        "Name": "IP Webcam # 10.20.3.110:8080",  
        "Extension": null  
    }  
}  
  
NetworkProtocols :  
{  
    "NetworkProtocols": [  
        {"Name": "HTTP",  
         "Enabled": true,  
         "Port": 8080,  
         "Extension": null  
     }  
]  
}  
  
Device Information:  
{  
    "Manufacturer": "Pavel Khlebovich",  
    "Model": "IP Webcam",  
    "FirmwareVersion": "B49",  
    "SerialNumber": "NNNNNNNN",  
    "HardwareID": "Android Device"  
}
```

GetUser,
GetDiscoveryMode,
GetSystemDateAndTime

```
Users :  
{  
    "User": [  
        {  
            "Username": "admin",  
            "Password": "admin",  
            "UserLevel": "Administrator",  
            "Extension": null  
        }  
    ]  
}  
  
DiscoveryMode :  
{  
    "DiscoveryMode": "Discoverable"  
}  
  
System Date and Time :  
{  
    "SystemDateAndTime": {  
        "DateTimeType": "Manual",  
        "DaylightSavings": false,  
        "TimeZone": {  
            "ID": "+02:00"  
        },  
        "UTCDateTime": {  
            "Time": {  
                "Hour": 5,  
                "Minute": 55,  
                "Second": 52  
            },  
            "Date": {  
                "Year": 2023,  
                "Month": 5,  
                "Day": 15  
            }  
        },  
        "localDateTime": {  
            "Time": {  
                "Hour": 5,  
                "Minute": 55,  
                "Second": 52  
            },  
            "Date": {  
                "Year": 2023,  
                "Month": 5,  
                "Day": 15  
            }  
        }  
    }  
}
```

Metadata

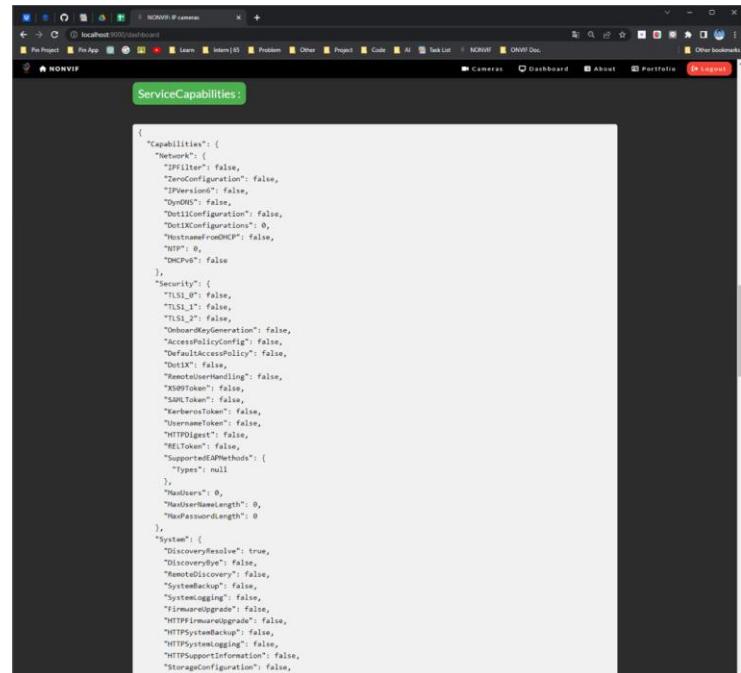
GetCapabilities



A screenshot of a web browser window titled "NONVIF IP cameras". The URL is "localhost:9000/dashboard". The page displays a JSON object under the heading "Capabilities":

```
{
  "capabilities": {
    "device": null,
    "device": [
      {
        "address": "http://10.20.3.110:8880/onvif/device_service",
        "network": {
          "IPfilter": false,
          "Zeroconfiguration": false,
          "IPv4": false,
          "IPv6": false,
          "DnSNS": false,
          "Extension": {
            "Dot11Configuration": false,
            "Extension": {
              "XMLName": {
                "Space": "",
                "local": ""
              }
            },
            "Attrs": null,
            "Children": null,
            "Text": ""
          }
        }
      }
    ],
    "system": {
      "DiscoveryResolve": true,
      "DiscoveryBy": false,
      "AnnounceDiscovery": true,
      "SystemBackup": false,
      "SystemLogging": false,
      "FirmwareUpgrade": false,
      "SupportedVersions": {
        "Major": 1,
        "Minor": 2
      },
      "Extension": {
        "HttpFirmwareUpgrade": false,
        "HttpsSystemBackup": false,
        "HttpsSystemLogging": false,
        "HttpsSupportInformation": false,
        "Extension": {
          "XMLName": {
            "Space": "",
            "local": ""
          }
        },
        "Attrs": null,
        "Children": null,
        "Text": ""
      }
    }
  }
}
```

GetServiceCapabilities

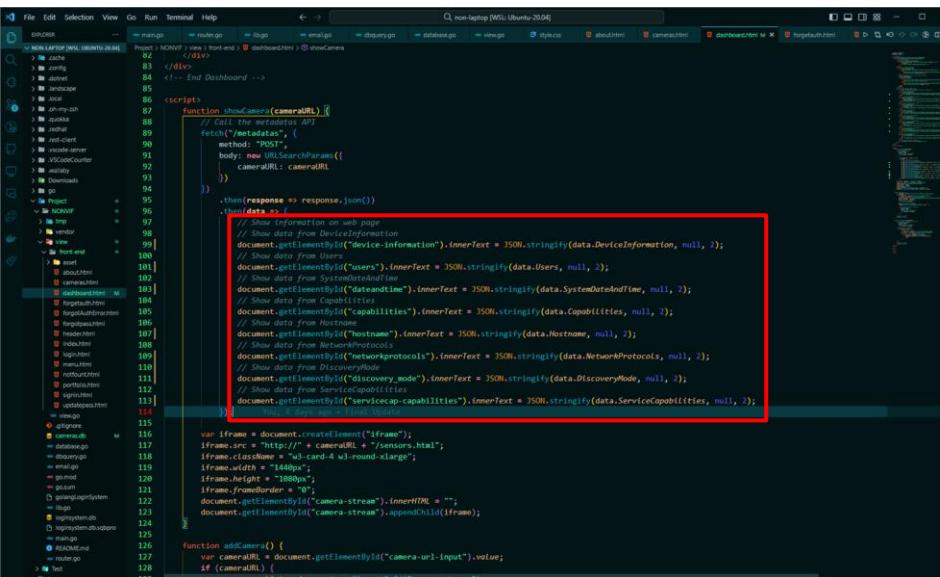


A screenshot of a web browser window titled "NONVIF IP cameras". The URL is "localhost:9000/dashboard". The page displays a JSON object under the heading "ServiceCapabilities":

```
{
  "capabilities": {
    "Network": {
      "IPfilter": false,
      "Zeroconfiguration": false,
      "IPversion": false,
      "DnSNS": false,
      "Dot11Configuration": false,
      "Dot1XConfigurations": 0,
      "HostnamedFromDHCP": false,
      "NTP": 0,
      "DHCPv6": false
    },
    "Security": {
      "TLS1": false,
      "TLS1_1": false,
      "TLS1_2": false,
      "DhOracleKeyGeneration": false,
      "AccessPolicyConfig": false,
      "DefaultAccessPolicy": false,
      "Dot1X": false,
      "RemoteUserHunting": false,
      "RemoteProvisioning": false,
      "SNTToken": false,
      "KerberosToken": false,
      "UsernameToken": false,
      "HTTPDigest": false,
      "RSToken": false,
      "SupportedSMPMethods": {
        "Type": null
      },
      "MaxUsers": 0,
      "MaxUserNameLength": 0,
      "MaxPasswordLength": 0
    },
    "System": {
      "DiscoveryResolve": true,
      "DiscoveryBy": false,
      "AnnounceDiscovery": false,
      "SystemBackup": false,
      "SystemLogging": false,
      "FirmwareUpgrade": false,
      "HttpFirmwareUpgrade": false,
      "HttpsSystemBackup": false,
      "HttpsSystemLogging": false,
      "HttpsSupportInformation": false,
      "StorageConfiguration": false
    }
  }
}
```

Metadata

การเขียน JS เรียกข้อมูลเมตาแบบเก่า

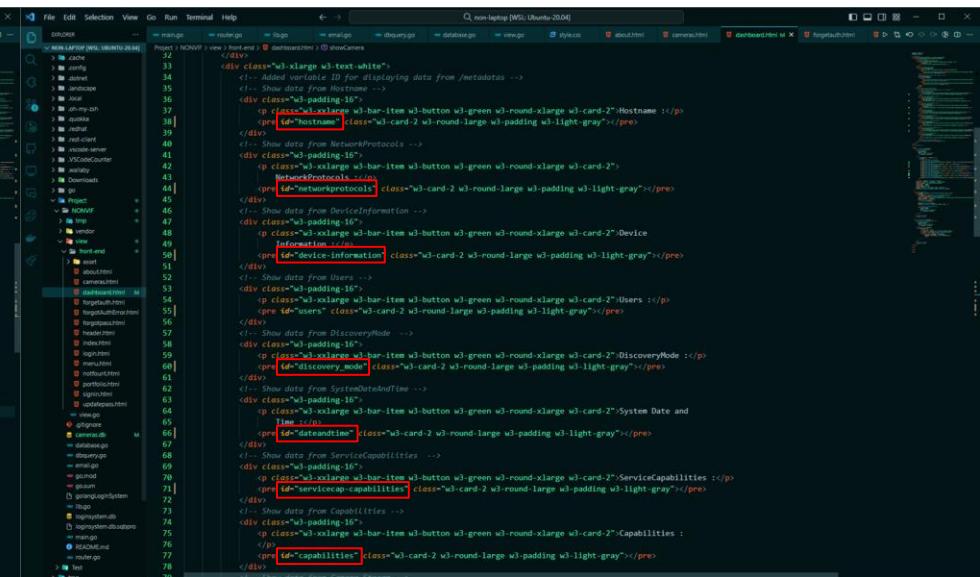


```
function showCameras(cameraURL) {
    // Call the metadata API
    fetch("/metadata", {
        method: "POST",
        body: new URLSearchParams({
            cameraURL: cameraURL
        })
    }).then(response => response.json())
    .then(data => {
        // Show information on web page
        // Show data from DeviceInformation
        document.getElementById("device-information").innerText = JSON.stringify(data.DeviceInformation, null, 2);
        // Show data from Users
        document.getElementById("users").innerText = JSON.stringify(data.Users, null, 2);
        // Show data from System
        document.getElementById("system").innerText = JSON.stringify(data.System, null, 2);
        // Show data from Capabilities
        document.getElementById("capabilities").innerText = JSON.stringify(data.Capabilities, null, 2);
        // Show data from Hostname
        document.getElementById("hostname").innerText = JSON.stringify(data.Hostname, null, 2);
        // Show data from NetworkProtocols
        document.getElementById("networkprotocols").innerText = JSON.stringify(data.NetworkProtocols, null, 2);
        // Show data from DiscoveryMode
        document.getElementById("discovery_mode").innerText = JSON.stringify(data.DiscoveryMode, null, 2);
        // Show data from Sensors
        document.getElementById("sensors").innerText = JSON.stringify(data.Sensors, null, 2);
        // Show data from ServiceCapabilities
        document.getElementById("servicecapabilities").innerText = JSON.stringify(data.ServiceCapabilities, null, 2);

        var frame = document.createElement("iframe");
        frame.src = "http://127.0.0.1:5000/sensors.html";
        frame.className = "w3-card-4 w3-round-large";
        frame.width = "1440px";
        frame.height = "1080px";
        frame.frameborder = "0";
        document.getElementById("camera-stream").innerHTML = "";
        document.getElementById("camera-stream").appendChild(frame);
    });
}

function addCamera() {
    var cameraURL = document.getElementById("camera-url-input").value;
    if (cameraURL) {
        showCameras(cameraURL);
    }
}
```

การเขียน HTML เรียกข้อมูลเมตาแบบเก่า



```
<!-- Added variable ID for displaying data from /metadata -->
<!-- Show data from Hostname -->


<p>Hostname :</p>
    <pre>{{#hostname}}</pre>


<!-- Show data from NetworkProtocols -->


<p>NetworkProtocols :</p>
    <pre>{{#networkprotocols}}</pre>


<!-- Show data from DeviceInformation -->


<p>Device Information :</p>
    <pre>{{#device-information}}</pre>


<!-- Show data from Users -->


<p>Users :</p>
    <pre>{{#users}}</pre>


<!-- Show data from DiscoveryMode -->


<p>Discovery Mode :</p>
    <pre>{{#discovery_mode}}</pre>


<!-- Show data from SystemDateAndTime -->


<p>System Date and Time :</p>
    <pre>{{#dateandtime}}</pre>


<!-- Show data from ServiceCapabilities -->


<p>ServiceCapabilities :</p>
    <pre>{{#servicecapabilities}}</pre>

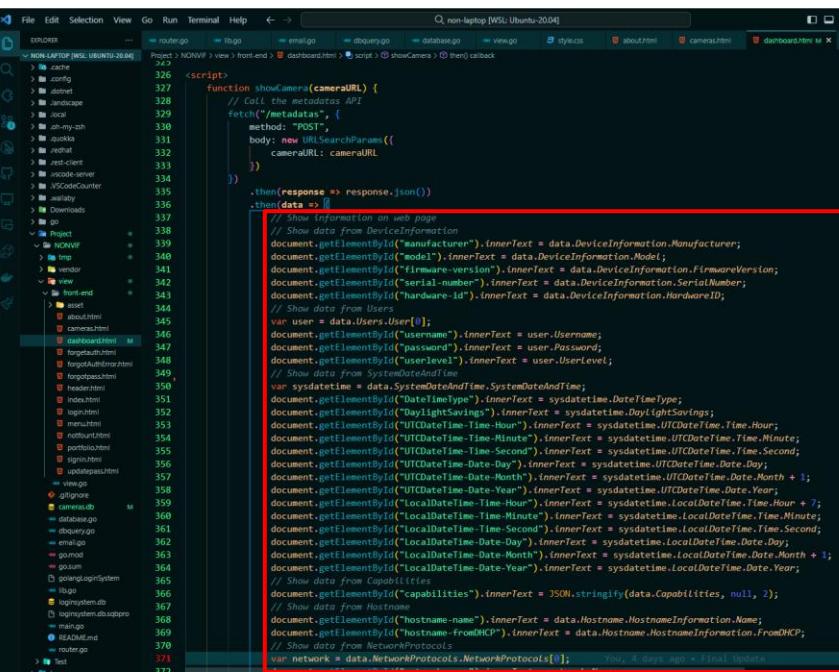

<!-- Show data from Capabilities -->


<p>Capabilities :</p>
    <pre>{{#capabilities}}</pre>


```

Metadata

การเขียน JS เรียกข้อมูลเมตาแบบใหม่



```
Project > NONVIF > view > front-end > dashboard.html > script > showCamera > then() callback
Project > NONVIF > view > front-end > dashboard.html.m > dashboard.html > script > showCamera > then() callback

<script>
    function showCamera(cameraURL) {
        // Call the metadatas API
        fetch('/metadatas', {
            method: "POST",
            body: new URLSearchParams({
                cameraURL: cameraURL
            })
        })
        .then(response => response.json())
        .then(data => {
            // Show data from DeviceInformation
            document.getElementById("manufacturer").innerText = data.DeviceInformation.Manufacturer;
            document.getElementById("model").innerText = data.DeviceInformation.Model;
            document.getElementById("firmware-version").innerText = data.DeviceInformation.FirmwareVersion;
            document.getElementById("serial-number").innerText = data.DeviceInformation.SerialNumber;
            document.getElementById("hardware-id").innerText = data.DeviceInformation.HardwareID;

            // Show data from User
            document.getElementById("username").innerText = user.Username;
            document.getElementById("password").innerText = user.Password;
            document.getElementById("userlevel").innerText = user.UserLevel;

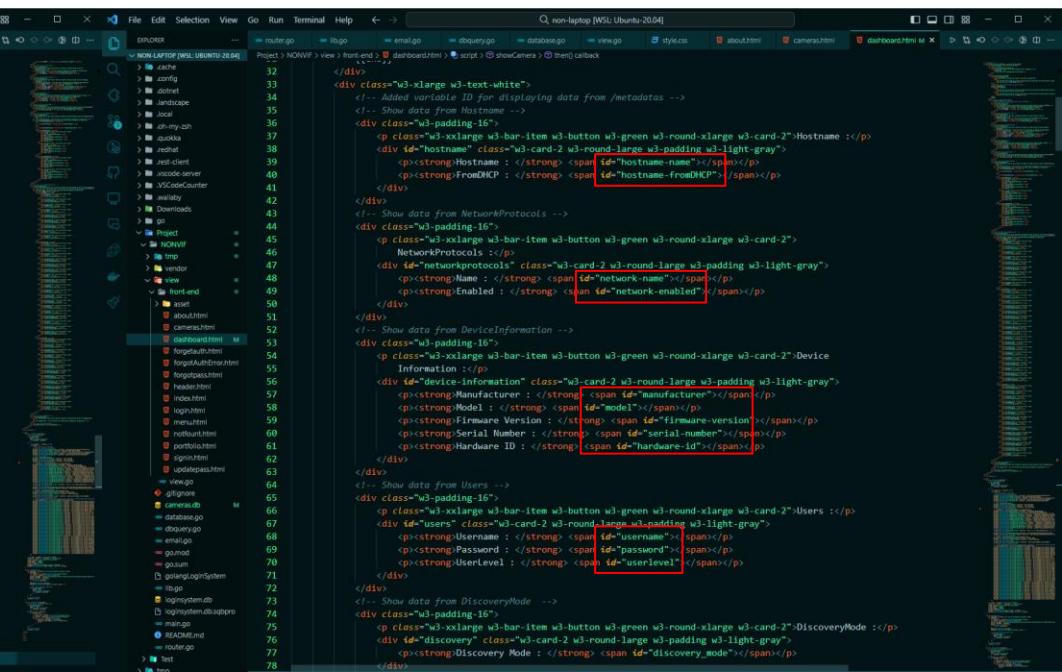
            // Show data from SystemDateAndTime
            var systime = data.SystemDateAndTime.SystemDateAndTime;
            document.getElementById("DateType").innerText = systime.DateType;
            document.getElementById("DaylightSavings").innerText = systime.DaylightSavings;
            document.getElementById("UTCDateTime-Time-Hour").innerText = systime.UTCDateTime.Time.Hour;
            document.getElementById("UTCDateTime-Time-Minute").innerText = systime.UTCDateTime.Time.Minute;
            document.getElementById("UTCDateTime-Time-Second").innerText = systime.UTCDateTime.Time.Second;
            document.getElementById("UTCDateTime-Date-Day").innerText = systime.UTCDateTime.Date.Day;
            document.getElementById("UTCDateTime-Date-Month").innerText = systime.UTCDateTime.Date.Month + 1;
            document.getElementById("UTCDateTime-Date-Year").innerText = systime.UTCDateTime.Date.Year;
            document.getElementById("LocalDateTime-Time-Hour").innerText = systime.LocalDateTime.Time.Hour + 7;
            document.getElementById("LocalDateTime-Time-Minute").innerText = systime.LocalDateTime.Time.Minute;
            document.getElementById("LocalDateTime-Time-Second").innerText = systime.LocalDateTime.Date.Second;
            document.getElementById("LocalDateTime-Date-Day").innerText = systime.LocalDateTime.Date.Day;
            document.getElementById("LocalDateTime-Date-Month").innerText = systime.LocalDateTime.Date.Month + 1;
            document.getElementById("LocalDateTime-Date-Year").innerText = systime.LocalDateTime.Date.Year;

            // Show data from Capabilities
            document.getElementById("capabilities").innerText = JSON.stringify(data.Capabilities, null, 1);

            // Show data from Hostname
            document.getElementById("hostname-name").innerText = data.Hostname.HostnameInformation.Name;
            document.getElementById("hostname-fromDHCP").innerText = data.Hostname.HostnameInformation.FromDHCP;

            // Show data from NetworkProtocols
            var network = data.NetworkProtocols.NetworkProtocols[1];
        });
    }
</script>
```

การเขียน HTML เรียกข้อมูลเมตาแบบใหม่



```
Project > NONVIF > view > front-end > dashboard.html > script > showCamera > then() callback
Project > NONVIF > view > front-end > dashboard.html.m > dashboard.html > script > showCamera > then() callback

<div class="w3-xlarge w3-text-white">
    <!-- Added variable ID for displaying data from /metadatas -->
    <!-- Show data From Hostname -->
    <div class="w3-padding-16">
        <p class="w3-xlarge w3-bar-item w3-button w3-green w3-round-xlarge w3-card-2">Hostname :</p>
        <div id="hostname" class="w3-card-2 w3-round-large w3-padding w3-light-gray">
            <p><strong>Hostname :</strong> <span id="hostname-name"></span></p>
            <p><strong>FromDHCP :</strong> <span id="hostname-FromDHCP"></span></p>
        </div>
    </div>
    <!-- Show data From NetworkProtocols -->
    <div class="w3-padding-16">
        <p class="w3-xlarge w3-bar-item w3-button w3-green w3-round-xlarge w3-card-2">NetworkProtocols :</p>
        <div id="networkProtocols" class="w3-card-2 w3-round-large w3-padding w3-light-gray">
            <p><strong>Network Name :</strong> <span id="network-name"></span></p>
            <p><strong>Enabled :</strong> <span id="network-enabled"></span></p>
        </div>
    </div>
    <!-- Show data From DeviceInformation -->
    <div class="w3-padding-16">
        <p class="w3-xlarge w3-bar-item w3-button w3-green w3-round-xlarge w3-card-2">Device Information :</p>
        <div id="deviceInformation" class="w3-card-2 w3-round-large w3-padding w3-light-gray">
            <p><strong>Manufacturer :</strong> <span id="manufacturer"></span></p>
            <p><strong>Model :</strong> <span id="model"></span></p>
            <p><strong>Firmware Version :</strong> <span id="firmware-version"></span></p>
            <p><strong>Serial Number :</strong> <span id="serial-number"></span></p>
            <p><strong>Hardware ID :</strong> <span id="hardware-id"></span></p>
        </div>
    </div>
    <!-- Show data From Users -->
    <div class="w3-padding-16">
        <p class="w3-xlarge w3-bar-item w3-button w3-green w3-round-xlarge w3-card-2">Users :</p>
        <div id="users" class="w3-card-2 w3-round-large w3-padding w3-light-gray">
            <p><strong>Username :</strong> <span id="username"></span></p>
            <p><strong>Password :</strong> <span id="password"></span></p>
            <p><strong>UserLevel :</strong> <span id="userlevel"></span></p>
        </div>
    </div>
    <!-- Show data From DiscoveryMode -->
    <div class="w3-padding-16">
        <p class="w3-xlarge w3-bar-item w3-button w3-green w3-round-xlarge w3-card-2">DiscoveryMode :</p>
        <div id="discovery" class="w3-card-2 w3-round-large w3-padding w3-light-gray">
            <p><strong>Discovery Mode :</strong> <span id="discovery_mode"></span></p>
        </div>
    </div>
```

Metadata

GetHostname, GetUser,
GetNetworkProtocols,
GetDeviceInformation

The screenshot shows the NORNIF IP Camera dashboard with several sections displaying device information:

- Hostname:** Hostname: IP Webcam @ 10.20.3.110:8080
FromDHCP: false
- NetworkProtocol:** Name: RTSP
Enabled: true
- Device Information:** Manufacturer: Pavel Khlebovich
Model: IP Webcam
Firmware Version: 849
Serial Number: SN000000
Hardware ID: Android Device
- Users:** Username: admin
Password: admin
UserLevel: Administrator

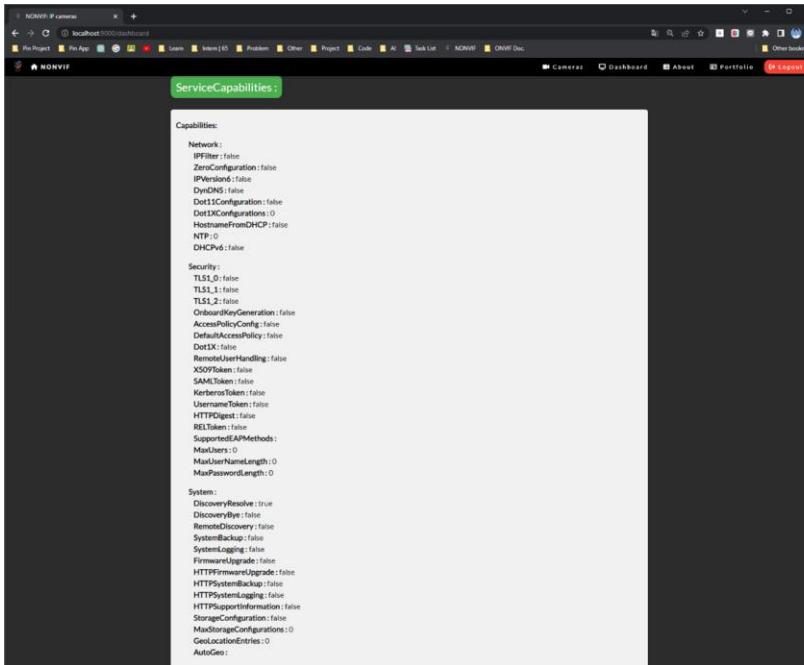
GetDiscoveryMode,
GetSystemDateAndTime,
GetServiceCapabilities

The screenshot shows the NORNIF IP Camera dashboard with sections for system and service capabilities:

- DiscoveryMode:** Discovery Mode: Discoverable
- System Date and Time:** SystemDateTimeType: Manual
DaylightSavings: false
UTCDateTime:
 - Time: 9:19:27 Minute:31 Second
 - Date: 13 Day 6 Month 2023 Year
LocalDateTime:
 - Time: 16:19:27 Minute:31 Second
 - Date: 13 Day 6 Month 2023 Year
- ServiceCapabilities:** Capabilities:
 - Network:** IPFilter: false
ZeroConfiguration: false
IPVersion6: false
DHCP: false
Dot1Configuration: false
Dot2Configurations: 10
HostsFromDHCP: false
NTP: false
DHCPv6: false
 - Security:** TLS1_0: false
TLS1_1: false
TLS1_2: false
OnboardKeyGeneration: false
AsymmetricKeySharing: false
DefaultAccessPolicy: false
Dot3: false
RemoteUserHandling: false

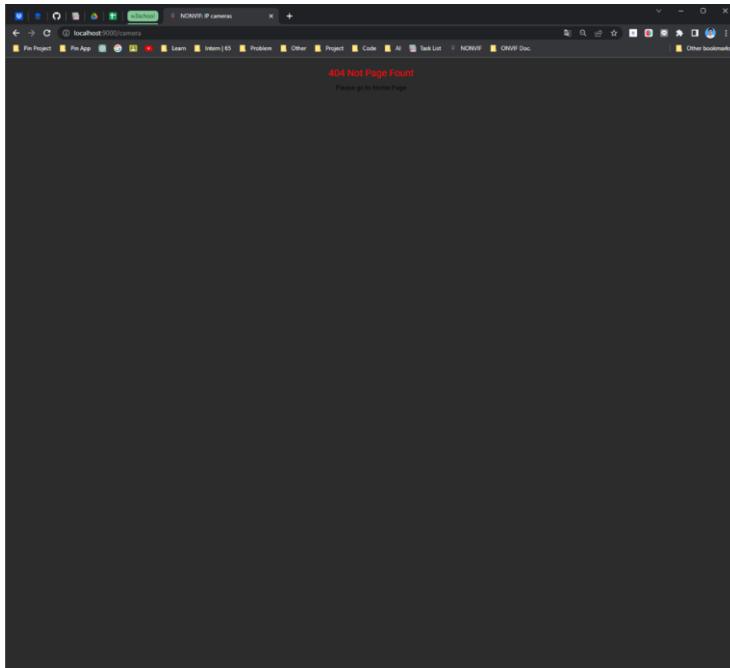
Metadata

GetServiceCapabilities



Other

404 Not Fount



สรุปผลการทดสอบ



เป็นไปตามวัตถุประสงค์

การจัดการกล้องไอพี
การสตรีมวิดีโอและ
การแสดงค่าข้อมูลเมตา



ข้อจำกัด

Golang ไม่มี Library ของ
ONVIF แบบ Official



ระยะเวลา

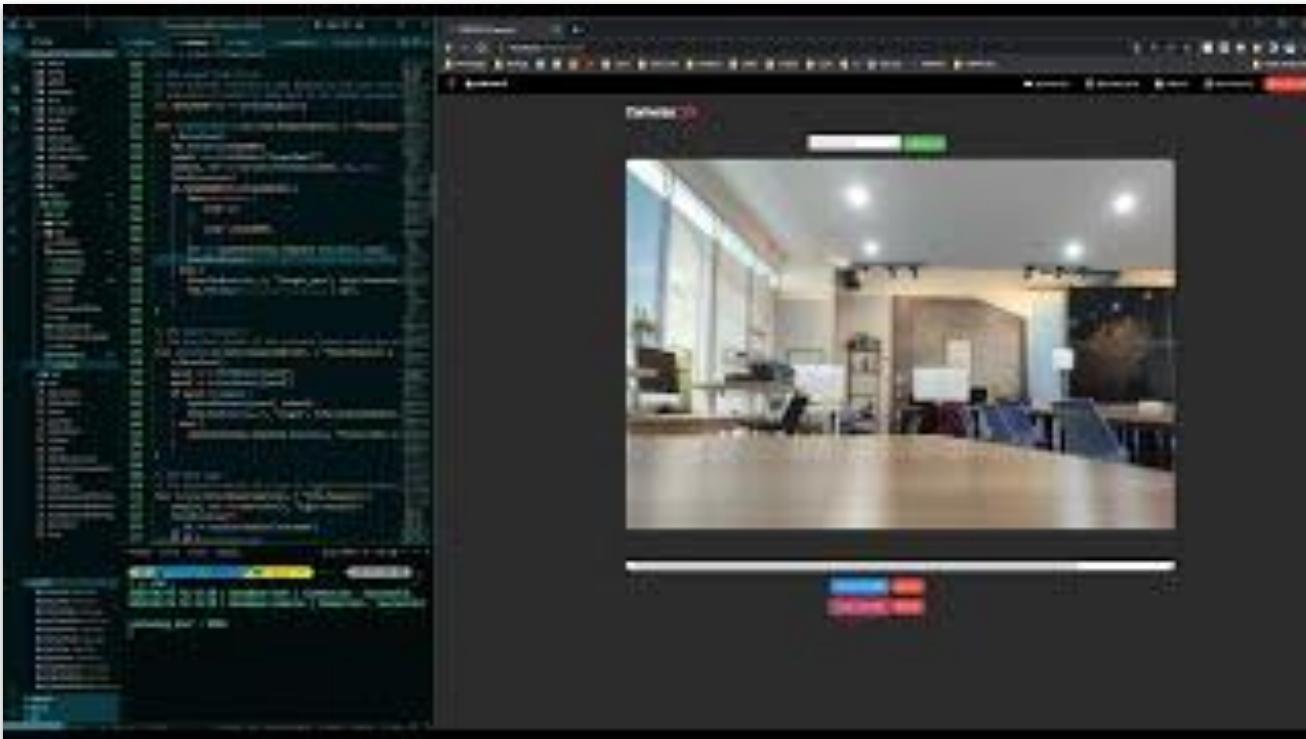
ระยะเวลาในการศึกษา
และพัฒนา



การพัฒนาต่อ

ใช้ go-opencv
หรือ gocv

Demo Video



Presented by



Nuntakorn : 62050553
Embedded Systems Engineering
Burapha University



**Thank for
Watching**