

**Ho Chi Minh University of Technology
Faculty of Computer Science and Engineering**



**SOFTWARE ENGINEERING
Project report #4**

Module Interface, Class Diagram, Sequence Diagram, Activity Diagram, Demo

Student's Name: Nguyen Sy Duc
Student's ID: 1752015
Project: Smart Parking

List of Contents

1. Module Interface
2. Class Diagram
3. Sequence Diagram
4. Activity Diagram
5. Working demonstration

1. Module Interface

In my project, I use two main interface

Figure 1 shows the module interface for Device

<<interface>> Device
+ getCurrentState(): int + getLastConfigure(): Datetime + saveConfigDate(Datetime date): void + disconnectServer(): int + connectServer(): int + reset(): void + setDeviceID(): void + getDeviceID(): String

Figure1

Figure 2 shows the module interface for Camera View Page

<<interface>> ICameraView
+ setController(CameraController): void + addCameraToGrid(Camera camera): + chooseCamera(): void + chooseRecord(): void + displayCameraScreen(): void + configCameraInGrid(Camera): void + removeCameraInGrid(Camera): void + setSelectedCameraInGrid(Camera): void + getIDSelectedCameraInGrid(): String + clearGrid(Camera camera):

Figure2

2. Class Diagram

Class diagram for Camera System

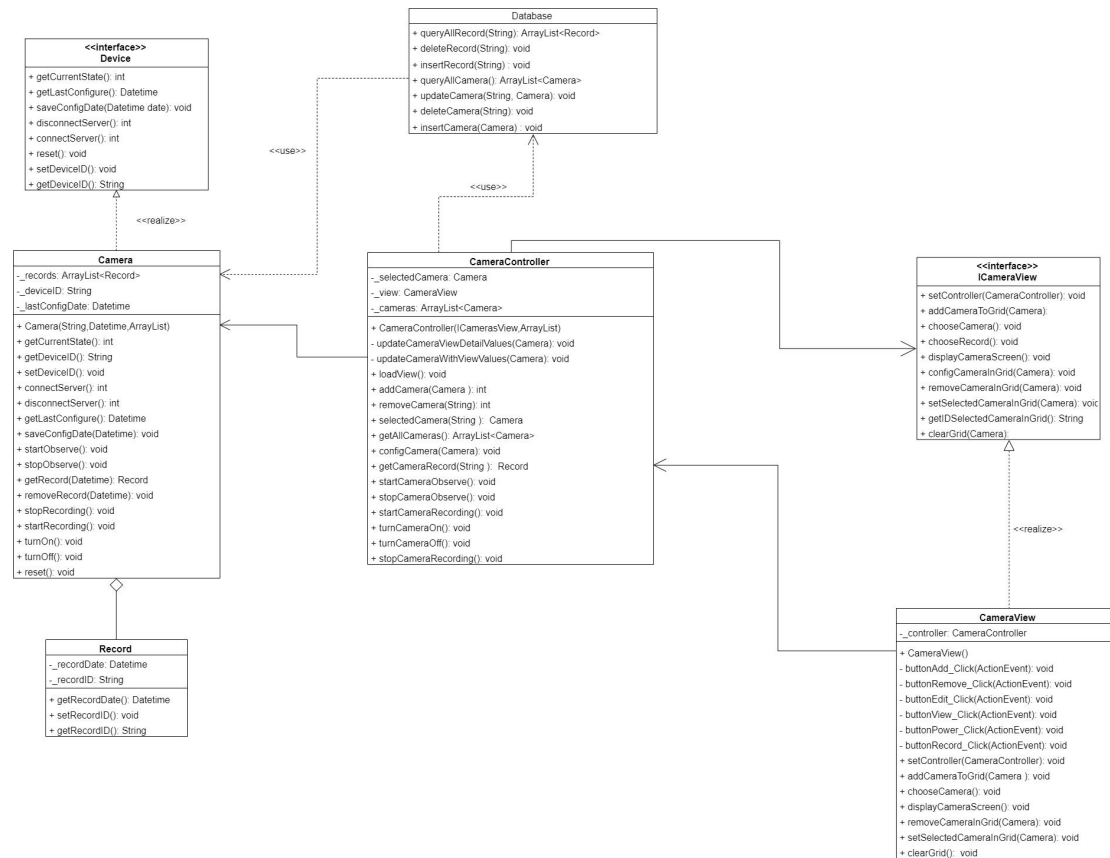


Figure3

Method Description:

Class Camera:

Method	Access Modifier	Parameter	ReturnType	Purpose
Camera	public	ID,LastConfig,List of records	No return	Create Camera
getCurrentState()	public		int	Return the state of camera
getDeviceID()	public		String	Return CameraID
setDeviceID()	public		Void	Set ID for Camera
connectServer()	public		int	Connect Camera to Server and Return the result
disconnectServer()	public		Void	Disconnect Camera from server
getLastConfigure()	public		Datetime	Return the last config date
startObserve()	Public		Void	Observe the view from the camera
stopObserve()	Public		Void	Stop Observing the camera

Method	Access Modified	Parameter	Return Type	Purpose
getRecord()	public	Datetime	Record	Return the record of that camera on that day
removeRecord()	public	Datetime	Void	Remove the record of that camera on that day
startRecording()	Public		Void	Start recording
stopRecording()	Public		Void	Stop recording and save the record
turnOn()	Public		Void	Turn on the camera
turnOff()	Public		Void	Turn off the camera
reset()	Public		Void	Reset the config

Class Record:

Method	Access Modified	Parameter	Return Type	Purpose
getRecordDate()	public		Datetime	Get the time when we recorded that record
setRecordID()	public		Void	Set Record ID
getRecordID()	public		String	Return the recordID

Class CameraController:

Method	Access Modified	Parameter	Return Type	Purpose
CameraController()	public	ICameraView, List Cameras	No return type	Create CameraController
updateCameraViewDetailValues()	private	Camera	void	Update the field text with the camera details
updateCameraWithViewValues()	private	Camera	Void	Save all Camera in the view to the Camera List
loadView()	Public		void	UpdateView
addCamera()	Public	Camera	Int	Add camera and return result
removeCamera()	Public	Camera	Int	Add camera and return result
selectedCamera()	Public	Camera	Camera	Get selected Camera on View
getAllCameras()	Public		ArrayList<Camera>	Query all cameras on Database
configCamera()	public	Camera	void	Edit the configuration of Camera
getCameraRecord()	public	RecordID	Record	Get the record
startCameraObserve()	Public		Void	Observe the view from the camera
stopCameraObserve()	Public		Void	Stop Observing the camera
startCameraRecording()	Public		Void	Start recording
stopCameraRecording()	Public		Void	Stop recording and save the record
turnCameraOn()	Public		Void	Turn on the camera
turnCameraOff()	Public		Void	Turn off the camera

Class CameraView:

Method	Access Modified	Parameter	Return Type	Purpose
CameraView()	public		No return type	Create CameraVlew
btnAdd_Click()	Private	ActionEvent	void	Call function addCamera() of controller to handle click Add button event
btnRemove_Click()	Private	ActionEvent	void	Call function removeCamera() of controller to handle click Remove button event
btnEdit_Click()	Private	ActionEvent	void	Call function configCamera() of controller to handle click Edit button event
btnPower_Click()	Private	ActionEvent	void	Call function turnCameraOn() / turnCameraOff() of controller to handle click Power button event
btnRecord_Click()	Private	ActionEvent	void	Call function startCameraRecording()/ stopCameraRecording()
setController()	public	CameraController	void	Choose Controller for Camera
chooseCamera()	public		void	Choose camera in the grid
addCameraToGrid()	public	Camera	void	Add Camera info to the grid
removeCameraInGrid()	public	Camera	void	Remove Camera in the grid
displayCameraScreen()	public		void	Display the Vision of the Camera
setSelectedCameraInGrid()	public		void	Highlight Camera In Grid
clearGrid()	public		void	Clear the Grid

Class Database:

Method	Access Modified	Parameter	Return Type	Purpose
queryAllCamera()	public		ArrayList<Camera>	Query from database all the Cameras' Info
insertCamera()	Public	Camera	void	Insert Camera to database
deleteCamera()	Public	CameraID	void	Delete Camera in database
updateCamera()	Public	Camera	void	Update Camera in database
queryAllRecord()	public	CameraID	ArrayList<Record>	Query from database all the Cameras' Record
insertCamera()	Public	Record	void	Insert Record to database
deleteCamera()	Public	RecordID	void	Delete Record in database

3. Sequence Diagram

Sequence Diagram for use-case “View Cameras”

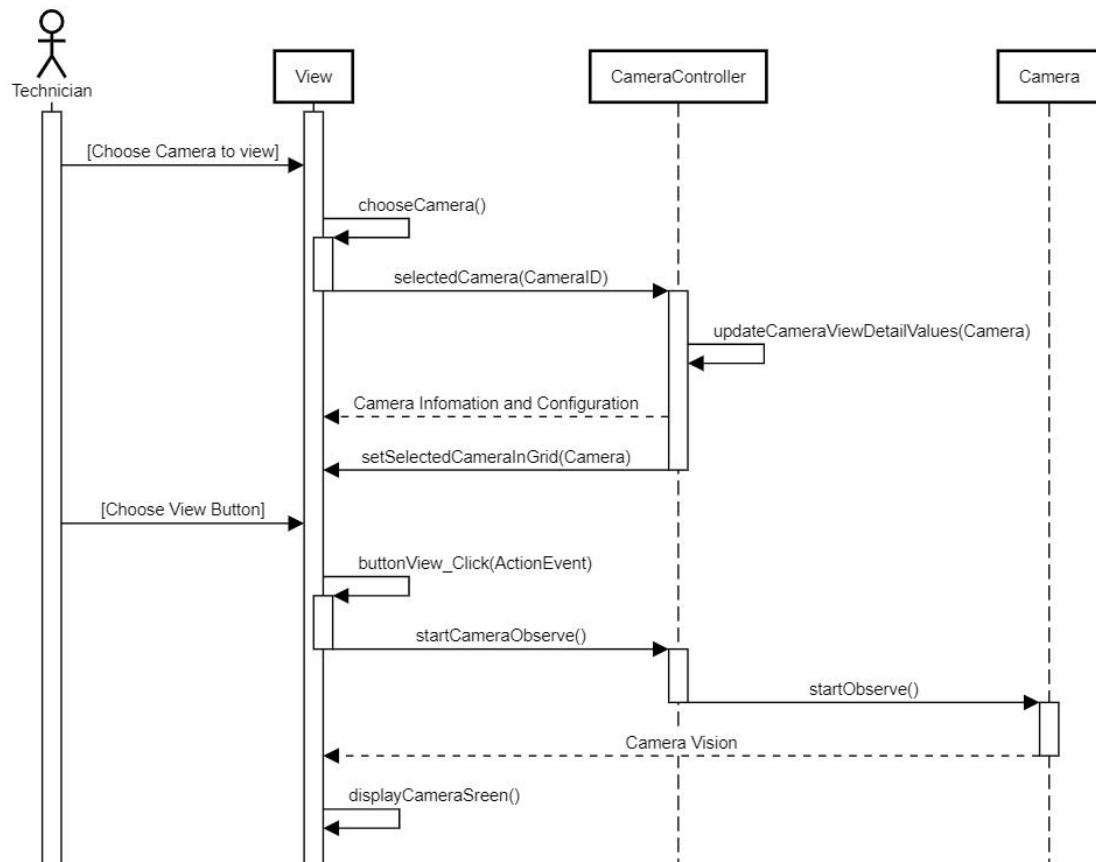


Figure4

4. Activity Diagram

Activity Diagram for task of viewing the camera of Technician

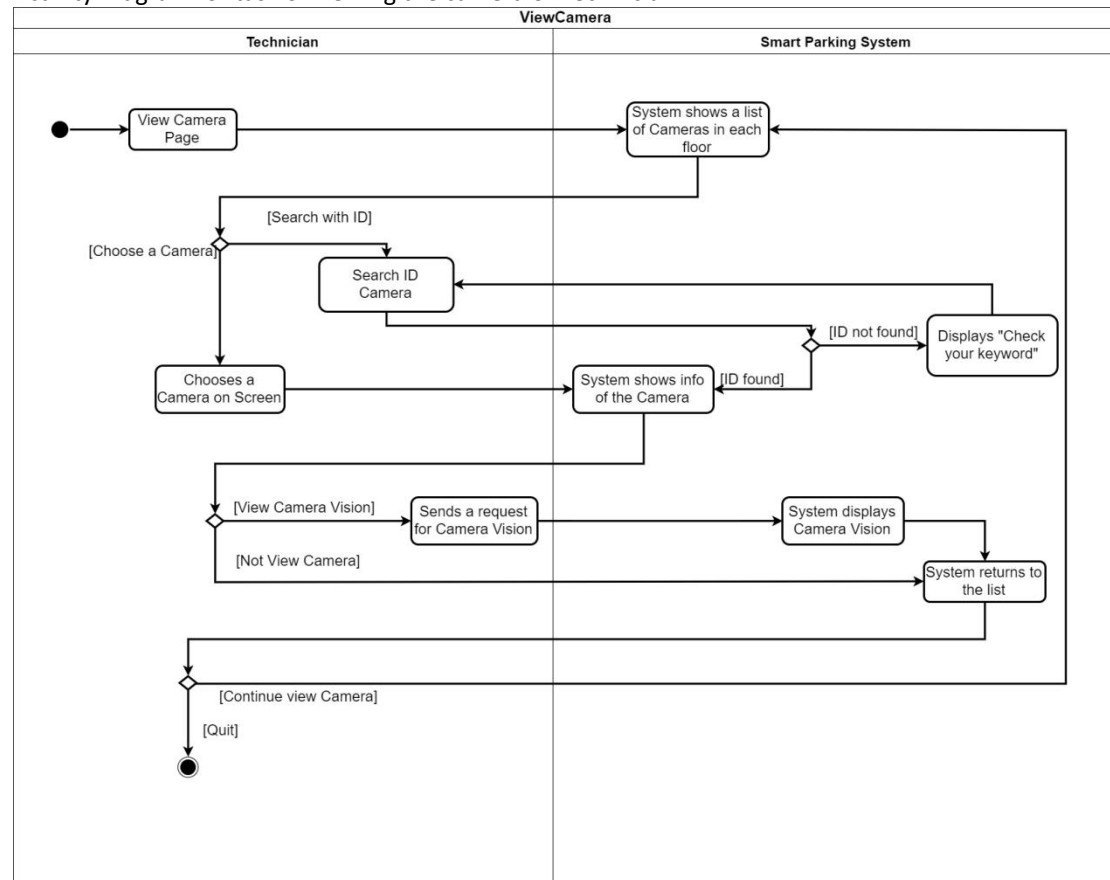


Figure5

5. Design Pattern

In my project, I use Model-View-Controller design Pattern.

Model is class Camera

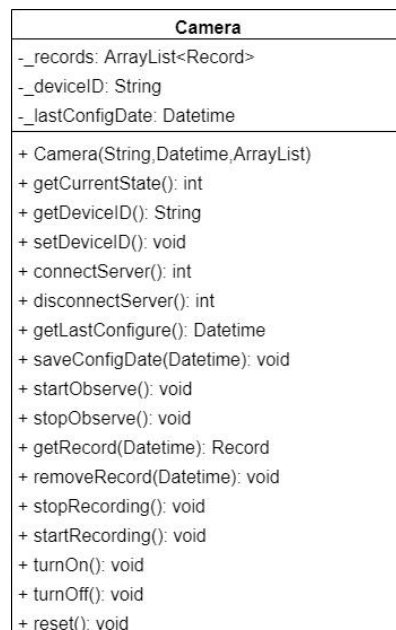


Figure6

View is class CameraView

CameraView
- _controller: CameraController
+ CameraView() - buttonAdd_Click(ActionEvent): void - buttonRemove_Click(ActionEvent): void - buttonEdit_Click(ActionEvent): void - buttonView_Click(ActionEvent): void - buttonPower_Click(ActionEvent): void + setController(CameraController): void + addCameraToGrid(Camera camera): + chooseCamera(): void + chooseRecord(): void + displayCameraScreen(): void + configCameraInGrid(Camera): void + removeCameraInGrid(Camera): void + setSelectedCameraInGrid(Camera): void + getIDSelectedCameraInGrid(): String + clearGrid(Camera camera):

Figure7

Controller is class CameraController

CameraController
- _selectedCamera: Camera - _view: CameraView - _cameras: ArrayList<Camera>
+ CameraController(ICamerasView, ArrayList) - updateCameraViewDetailValues(Camera): void - updateCameraWithViewValues(Camera): void + loadView(): void + addCamera(Camera camera): int + selectedCamera(String deviceId): Camera + getAllCameras(): ArrayList<Camera> + configCamera(Camera): void + getCameraRecord(String RecordID): Record + startCameraObserve(): void + stopCameraObserve(): void + startCameraRecording(): void + turnCameraOn(): void + turnCameraOff(): void + stopCameraRecording(): void

Figure8



I applied the Model-View-Controller because it is a well-proven design pattern to solve the problem of separating data (model) and user interface (view) concerns, so that changes to the user interface do not affect the data handling, and that the data can be changed without impacting/changing the UI.

6. Working Demonstration

Screen Flow for editing Camera:

https://

CameraID



FloorID F1

CameraID

Config

Add

Remove

Edit

View



Floor 1

CameraID	CurrentState	Configuration	LastConfig
F1A1	<div>OK</div>	Some CameraConfig	15 Sep, 8:56 AM (2013)
F1A2	<div>OK</div>	Some CameraConfig	15 Sep, 7:12 AM (2013)
F1A3	<div>Suspend</div>	Some CameraConfig	15 Sep, 4:34 AM (2013)
F1A4	<div>OK</div>	Some CameraConfig	15 Sep, 2:08 AM (2013)
F1A4	<div>Suspend</div>	Some CameraConfig	15 Sep, 8:56 AM (2013)

Choose the Camera to Edit

https://

CameraID



FloorID F1

CameraID F1A1

Config Some CameraConfig

Add

Remove

Edit

View

Floor 1

CameraID	CurrentState	Configuration	LastConfig
F1A1	<div>OK</div>	Some CameraConfig	15 Sep, 8:56 AM (2013)
F1A2	<div>OK</div>	Some CameraConfig	15 Sep, 7:12 AM (2013)
F1A3	<div>Suspend</div>	Some CameraConfig	15 Sep, 4:34 AM (2013)
F1A4	<div>OK</div>	Some CameraConfig	15 Sep, 2:08 AM (2013)
F1A4	<div>Suspend</div>	Some CameraConfig	15 Sep, 8:56 AM (2013)

When we click on the Camera. It will display the camera info to the field text.

Then we change the Config.

https://

CameraID

FloorID

F1

CameraID

F1A1

Config

New Config

Add

Remove

Edit

View

Floor 1

CameraID	CurrentState	Configuration	LastConfig
F1A1	OK	Some CameraConfig	15 Sep, 8:56 AM (2013)
F1A2	OK	Some CameraConfig	15 Sep, 7:12 AM (2013)
F1A3	Suspend	Some CameraConfig	15 Sep, 4:34 AM (2013)
F1A4	OK	Some CameraConfig	15 Sep, 2:08 AM (2013)
F1A4	Suspend	Some CameraConfig	15 Sep, 8:56 AM (2013)

Then we click Edit

https://

CameraID

FloorID

F1

CameraID

F1A1

Config

New Config

Add

Remove

Edit

View

Floor 1

CameraID	CurrentState	Configuration	LastConfig
F1A1	OK	Some CameraConfig	15 Sep, 8:56 AM (2013)
F1A2	OK	Some CameraConfig	15 Sep, 7:12 AM (2013)
F1A3	Suspend	Some CameraConfig	15 Sep, 4:34 AM (2013)
F1A4	OK	Some CameraConfig	15 Sep, 2:08 AM (2013)
F1A4	Suspend	Some CameraConfig	15 Sep, 8:56 AM (2013)

Then it will update in database and in the view.

https://

CameraID

FloorID

F1

CameraID

F1A1

Config

New Config

Add

Remove

Edit

View

Floor 1

CameraID	CurrentState	Configuration	LastConfig
F1A1	OK	New Config	15 Sep, 9:00 AM (2013)
F1A2	OK	Some CameraConfig	15 Sep, 7:12 AM (2013)
F1A3	Suspend	Some CameraConfig	15 Sep, 4:34 AM (2013)
F1A4	OK	Some CameraConfig	15 Sep, 2:08 AM (2013)
F1A4	Suspend	Some CameraConfig	15 Sep, 8:56 AM (2013)

Screen Flow for Viewing the Camera:

We choose the camera first then click View button.

https://

CameraID

FloorID

F1

CameraID

F1A1

Config

New Config

Add

Remove

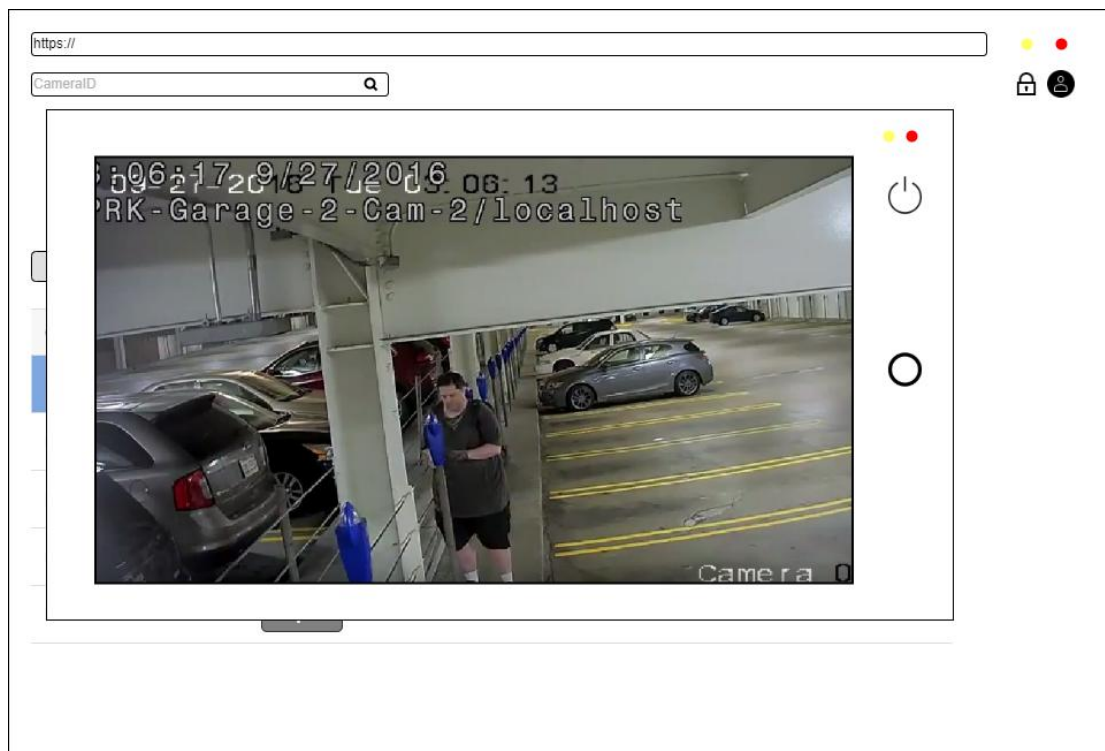
Edit

View

Floor 1

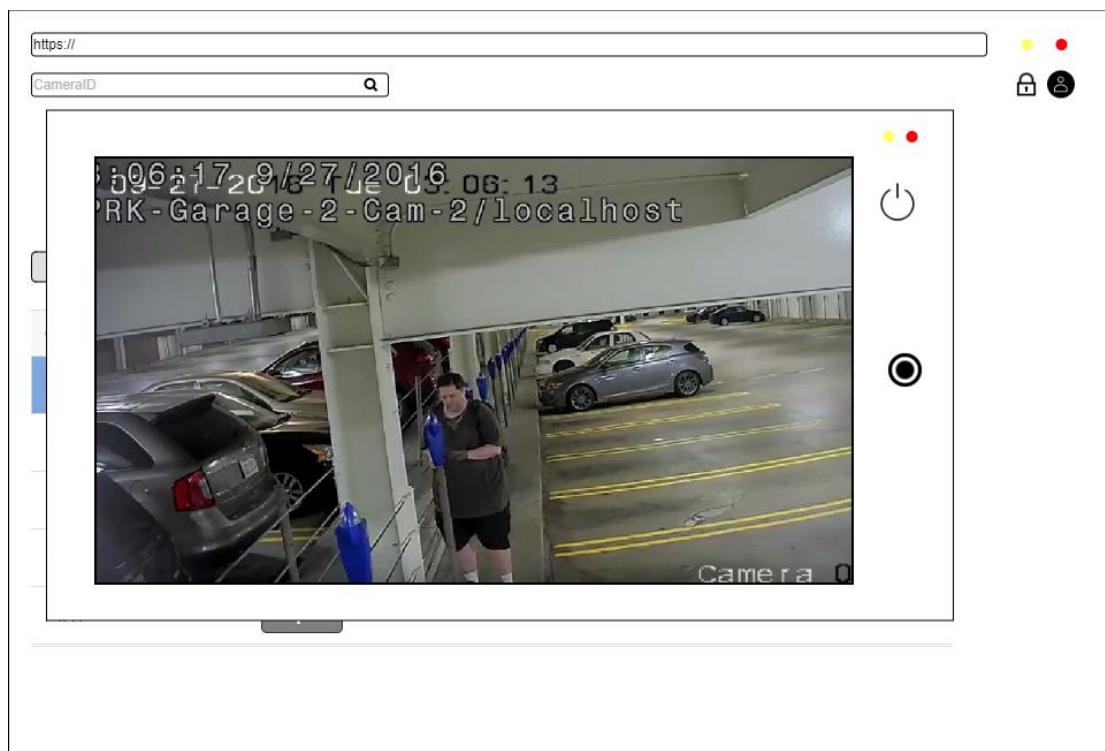
CameraID	CurrentState	Configuration	LastConfig
F1A1	OK	New Config	15 Sep, 9:00 AM (2013)
F1A2	OK	Some CameraConfig	15 Sep, 7:12 AM (2013)
F1A3	Suspend	Some CameraConfig	15 Sep, 4:34 AM (2013)
F1A4	OK	Some CameraConfig	15 Sep, 2:08 AM (2013)
F1A4	Suspend	Some CameraConfig	15 Sep, 8:56 AM (2013)

Then it will display the Camera Screen:



Screen FLOW for Recording :

Click the record button then Camera will record.



Click the record button again it will stop Recording.

