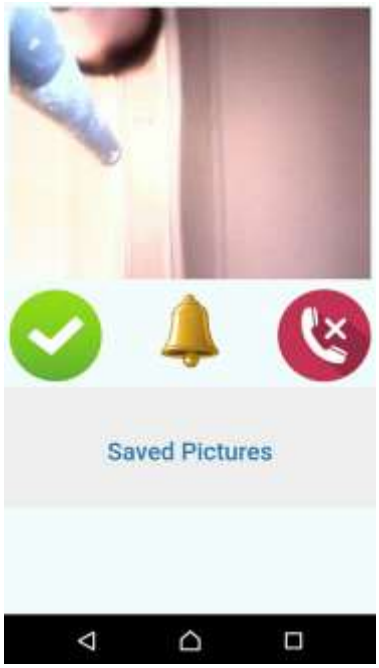
دریافت تصویر دوربین دیجیتال و انتقال بر بستر GPRS

جهت هدف‌مندتر شدن پرژه، سناریویی تحت عنوان ایفن هوشمند تعریف شده تا فرایند انجام ان طبق الگریتم خاصی نظام‌مند گردد. بدین منظور از این به بعد برد الکترنیکی که در نقش پنل جلوی درب طراحی شده و شامل ماژول دوربین و سیم‌کارت و همچنین سوییچ زنگ و دو LED بمنظور نشانگر رله میباشد، ایفن گفته میشود.

برد ایفن شامل میکرکنترلر ATmega48PA و سوییچ زنگ و نشانگر LED و ماژول GPRS

در این سناریو با فشرده شدن سوییچ، عمل عکس‌برداری انجام شده و بر بستر GPRS به سرور ارسال میگردد. همچنین فشرده‌شدن سوییچ توسط یک فلگ به سرور اطلاع داده میشود. سرور مورد استفاده که رابطی بین دستگاه ایفن و کاربر نهایی میباشد، تصویر دریافتی را جهت استفاده کاربر ذخیره کرده و کاربر میتواند با بهره‌گیری از صفحه وب و یا اپ اندریدی که در اختیارش قرار دارد، از فشرده شدن سوییچ زنگ مطلع شده و تصویر فرد را روی صفحه ببیند و توسط دو کلید که روی اپ طراحی شده، درب را باز کرده و یا الارمی را فعال نماید.



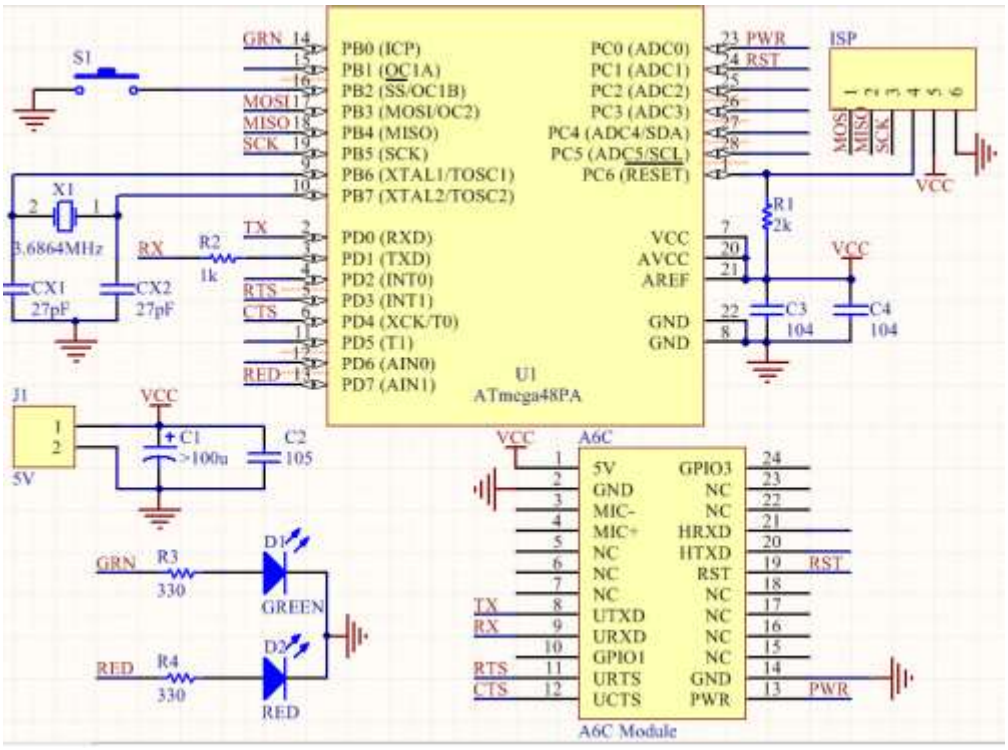
نمایی از اینترفیس طراحی شده

در بالای صفحه، تصویر دریافتی نشان داده میشود و با کلیک روی ان، فلگ دریافت تصویر جدید ست میشود.

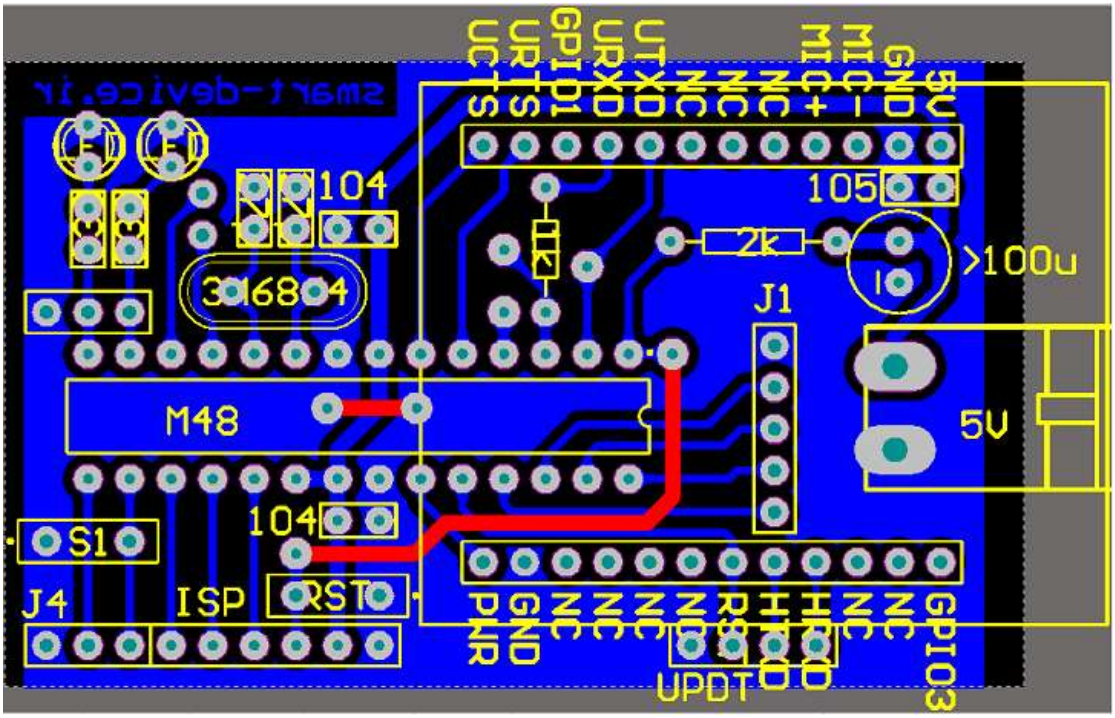
در زیر تصویر دریافتی، دو کلید بعنوان Accept(باز کردن درب) و Reject(الارم) و یک نمایشگر به شکل زنگ وجود دارد که فشرده شدن زنگ را نشان میدهد. با کلیک روی هر کدام از کلیدها، فلگ مربوط به فرمانش فعال شده و با کلیک روی نشانگر زنگ، صدای زنگ دریافتی و ویبره قطع میگردد.

در پایین صفحه هم لینک تصاویر دریافتی ذخیره شده وجود دارد که با کلیک روی ان میتوان به تصاویر دسترسی پیدا کرد. بدلیل محدودیت در فضای سرور، ذخیره تصاویر به 10 عدد محدود شده که بسته به میزان فضای موجود میتوان تعداد ذخیره‌سازی را افزایش داد.

جزییات پرژه: سخت‌افزار



شماتیک برد ایفن



پی‌سی‌بی طراحی شده در التیوم

میکرکنترلر مورد استفاده ATmega48PA و برنامه‌نویسی gcc تحت محیط ویژوال استادیو انجام گرفت که حدود 60% فلش و 50% رم میکرو اشغال شده.

برای اجرای پرژه، از ماژول A20 بهره گرفته شده است. البته در ابتدا بنا بود که A6C درایو شود که بدلیل اسیب دیدن ماژول، ادامه کار با A20 انجام شد.

هر دو ماژول ساخت شرکت AI Thinker و شامل ارتباط GSM/GPRS بوده و قادر به دریافت تصویر از دوربین OV7670 میباشند. ماژول A20 علاوه بر قابلیت‌های A6C، دارای یک میکروکنترلر ESP8266 جهت ارتباط وای‌فای میباشد که التبه در این پرژه مورد استفاده قرار نگرفته است.

مزیت ماژولهای این خانواده نسبت به ماژولهای SIM800، امکان اتصال مستقیم دوربین و دریافت دیتای دوربین و کمپرس کردن به فرمت JPG و ارسال سریع به وب میباشد.

تغذیه مدار توسط یک اداپتر شارژر اندریدی از طریق کانکتر میکرو USB تعبیه شده روی ماژول تامین میگردد. البته بدلیل نازک بودن سیمهای شارژرهای چینی ممکن است هنگام انتقال دیتا که ماژول جریان بیشتری میکشد، ولتاژ کاهش یابد که باید جهت جبران خازنی با ظرفیت بالا روی پایه تغذیه ماژول لحیم شود.

البته یک ترمینال هم برای ولتاژ ورودی در نظر گرفته شده که در صورت نیاز میتوان از ان بهره گرفت.

در نقاط مختلف برد خازنهایی جهت تثبیت ولتاژ و حذف ریپل درنظر گرفته شده است.

فرکانس کریستال میکرو3.6864MHz میباشد تا خطای ارتباط یوارت صفر شود.

دو LED به رنگهای سبز و قرمز از طریق مقاومت 330 به پایه های PB0 و PD7 وصل شده اند که بعنوان خروجی نمایشگر رله های بازکردن در و یا بصدا دراوردن الارم میباشد.

میکروسوییچی که به پایه PB2 وصل شده، بعنوان ورودی شاسی زنگ بکار میرود.

کانکتر ISP جهت پرگرم کردن میکرو میباشد و پایه ریست میکرو هم با یک مقاومت 2کیلو پول اپ شده است.

سعی شده پایه های بیکار میکرو بیرون کشیده شود تا درصورت نیاز بتوان از انها بهره گرفت.

ارتباط میکرو با ماژول از طریق دو پایه RX و TX میباشد. از انجا که تغذیه میکرو 5ولت است و تغذیه ماژول 3.3ولت میباشید، پایه TX میکرو با یک مقاومت 1کیلو به RX ماژول وصل شده تا ازاسیب احتمالی ورودی ماژول جلوگیری شود. پایه RST ماژول هم به پایه PC1 میکرو وصل شده تا درصورت نیاز میکرو بتواند ماژول را ریست سخت افزاری نماید. البته پایه های PWR و CTS و RTS ماژول نیز به میکرو وصل شده تا درصورت نیاز امکان دسترسی وجود داشته باشد.

ارتباط میکرکنترلر با ماژول از طریق دستورات ATCommand یوارت بشرح زیر میباشد:

جهت تنظیم بادریت اتومات(که اینجا 9600 است) و چک کردن اماده بکار بودن ماژول	**AT**
غیرفعال کردن اکوی ماژول	**ATE0**
چک کردن رجیستر شدن سیمکارت	**AT+CREG?**
خاموش کردن پیامهای ماژول(بخاطر ایجاد مزاحمت پیامهای اپراترها در کارکرد سیستم)	**AT+CNMI=0,0,0,0,0**
تنظیم پرتکل و اکسسپوینت جهت ارتباط GPRS	**AT+CGDCONT=1,"IP","mtnirancell"**
تنظیم نام اکسسپوینت و یوزر/پسورد	**AT+CSTT="mtnirancell","",""**
برقراری ارتباط GPRS جهت اتصال به اینترنت	**AT+CIICR**
دریافت ایپی ادرس ماژول	**AT+CIFSR**
فعال کردن فلش دوربین جهت روشن شدن هنگام عکسبرداری	**AT+CAMCFG=0,1**

البته ماژول کانفیگهای بیشتری همچون دید در شب و چرخش و کیفیت تصویر و ... هم دارد که درصورت نیاز میتوان از انها بهره گرفت.

استارت دوربین با کیفیت VGA و رزولوشن x480640	**AT+CAMSTART=1**
کپچر تصویر	**AT+CAMCAP**

دریافت تصویر از طریق پرت سریال، البته در این پرژه از این دستور استفاده نشده ولی جهت ذخیره لوکال میتوان از ان بهره گرفت **AT+CAMRD**

ارسال تصویر به وب‌سرور اسمارت‌دیوایس و پرت 80 به روش پست **AT+CAMPOST="smart-device.ir/ic/up.php?n=0.jpg",80**

AT+CIPSTART="TCP","smart-device.ir",80

استارت سوکت تی‌سی‌پی به وب‌سرور روی پرت 80 جهت ارسال فلگ زنگ و چک کردن فلگهای اکسپت/ریجکت

شروع ارسال دیتا **AT+CIPSEND**

بعد از ارسال این فرمان، اگر در مرحله قبل سوکت موفق باشد ماژول کرکتر > را بر میگرداند و میکرو این دو خط را به ماژول ارسال میکند:

اگر زنگ خورده باشد، عدد 1 بعنوان پارامتر به هاست ارسال میشود **GET /ic/mhsn.php?i=1 HTTP/1.1**

معرفی هاست جهت برقراری ارتباط **Host: smart-device.ir**

سپس جهت اگاه‌سازی ماژول از اتمام ارسال دیتا، کرکتر اسکی با کد دسیمال **27**را ارسال نمود

درصورتیکه ارتباط موفقیت‌امیز باشد، سرور دیتای دریافتی را ذخیره کرده و فلگهای مربوط به اکسپت/ریجکت/عکس‌بردار را با فرمت **UUU1** و یا **UUU2** و یا **UUU3** برمیگرداند.

این دستور برای بستن سوکت بکار میرود **AT+CIPCLOSE**

درصورتیکه میکرو رشته **UUU1** و یا **UUU2** را دریافت نماید، پایه‌های مرتبط را که به دو ال‌ای‌دی سبز و قرمز متصل است فعال مینماید و درصورت دریافت **UUU3** وارد پرسه گرفتن عکس جدید و ارسال به سرور میگردد.

مدت زمان انتقال تصویر از ماژول به سرور، بستگی به انتن‌دهی محل و کیفیت کانکشن اینترنت داشته و در تستهایی که انجام گرفت، کمترین زمان حدود 10 ثانیه بدست امد.

جهت اجرای دستوراتی که در بالا گفته شد و دریافت و پردازش پاسخ ماژول توسط میکرو در بازه زمانیهای مشخص، از فلگ تایمر1 استفاده شده تا درصورت عدم دریافت پاسخ از ماژول کل سیستم ریست شود.

شرح برنامه میکرو

هدرهای مورد استفاده در ابتدای برنامه اینکلود شده اند:

```
#include <avr/io.h>
#include <avr/interrupt.h>
#include <avr/pgmspace.h>
#include <util/delay.h>
#include <string.h>
```

دو ماکرو که جهت فعال و غیرفعال کردن دریافت یوارت تعریف شده اند:

```
#define       rxEnable()    UCSR0B|=(1<<RXCIE0)|(1<<RXEN0)
#define       rxDisable()   UCSR0B&=~((1<<RXCIE0)|(1<<RXEN0))
```

رشته های ثابتی که جهت کاهش مصرف رم، در فلش میکرو تعریف شده اند:

```
const char AT[] PROGMEM = "AT\r\n";
const char ATE0[] PROGMEM = "ATE0\r\n";
const char CREG[] PROGMEM = "AT+CREG?\r\n";
const char CNMI[] PROGMEM = "AT+CNMI=0,0,0,0,0\r\n";
const char CAMCFG[] PROGMEM = "AT+CAMCFG=0,1\r\n";
const char CAMSTART[] PROGMEM = "AT+CAMSTART=1\r\n";
const char CRLF[] PROGMEM = "\r\n";
const char OK[] PROGMEM = "\r\nOK\r\n";
const char ERROR[] PROGMEM = "ERROR";
const char CGDCONT[] PROGMEM = "AT+CGDCONT=1,\"IP\",\"mtnirancell\"\r\n";
const char CSTT[] PROGMEM = "AT+CSTT=\"mtnirancell\",\"\",\"\"\r\n";
```

```
const char CIICR[] PROGMEM = "AT+CIICR\r\n";
const char CIFSR[] PROGMEM = "AT+CIFSR\r\n";
const char CONNECT[] PROGMEM = "CONNECT ";
const char RST[] PROGMEM = "AT+RST=1\r\n";
const char WEB[] PROGMEM = "AT+CIPSTART=\"TCP\",\"smart-device.ir\",80\r\n";
const char CLOSE[] PROGMEM = "AT+CIPCLOSE\r\n";
const char SEND[] PROGMEM = "AT+CIPSEND\r\n";
const char GET[] PROGMEM = "GET /ic/mhsn.php?i=";
const char HTTP[] PROGMEM = " HTTP/1.1\r\n";
const char HOST[] PROGMEM = "Host: smart-device.ir\r\n\r\n";
const char CODE1[] PROGMEM = "UUU1";
const char CODE2[] PROGMEM = "UUU2";
const char CODE3[] PROGMEM = "UUU3";
const char ATCAMCAP[] PROGMEM = "AT+CAMCAP\r\n";
const char CAMCAP[] PROGMEM = "CAMCAP:";
const char ATCAMPOST[] PROGMEM = "AT+CAMPOST=\"smart-device.ir/ic/up.php?n=0.jpg\",80\r\n";
const char CAMPOST[] PROGMEM = "+CAMPOST:";
const char CAMPOST0[] PROGMEM = "+CAMPOST:0";
const char CAMPOST1[] PROGMEM = "+CAMPOST:1";
const char NORESPONSE[] PROGMEM = "COMMAND NO RESPONSE!\r\n";
```

متغیرهای بکار رفته در برنامه:

```
volatile unsigned char rxBuff[256] = { 0 }, rxIndx = 0, pb2 = 1;
```

rxBuff ارایه بایتهای دریافتی از ماژول

rxIndx ایندکس بایت دریافتی

pb2 فلگ فشرده شدن زنگ

تابعی که برای ریست نرم افزاری میکرو بکار میرود:

```
void(*rstMicro) (void) = 0; // Reset uC by jumping to address 0
```

تابع راه اندازی اولیه:

```
void init(void) {
	DDRB |= 0x01;                          // خروجی دربازکن //
	PORTB |= 0x04;                                 // ورودی زنگ //
	DDRD |= 0x80;                        // خروجی الارم //
```

فعال کردن اینتراپت مربوط به پایه سوییچ زنگ(:PB2)

```
	PCICR  = (1<<PCIE0); PCMSK0 = (1<<PCINT2);      PCIFR  = (1<<PCIF0);
```

کانفیگ تایمر1 برای سرریز حدود 1 ثانیه:

```
	TCCR1B = (0 << ICNC1) | (0 << ICES1) | (0 << WGM13) | (0 << WGM12) | (0 << CS12) | (1 << CS11) | (1
<< CS10);     // ~1Hz@ 3.686400MHz
	TCNT1H = 0x00; TCNT1L = 0x00; ICR1H = 0x00; ICR1L = 0x00; OCR1AH = 0x00; OCR1AL = 0x00; OCR1BH =
0x00; OCR1BL = 0x00;
```

کانفیگ یوارت برای بادریت bps9600:

```
	UCSR0B = (1 << TXEN0);      UCSR0C = (1 << UCSZ01) | (1 << UCSZ00);
	UBRR0H = 0x00;       UBRR0L = 0x17;       // BaudRate=9600bps  @ 3.686400MHz
```

کانفیگ واچداگ جهت ریست کردن میکرو درصورت عدم ریست تایمر پس از حدود 4 ثانیه:

```
	WDTCSR = (0 << WDIF) | (0 << WDIE) | (1 << WDP3) | (1 << WDCE) | (1 << WDE) | (0 << WDP2) | (0 <<
WDP1) | (1 << WDP0);
```

```
	WDTCSR = (0 << WDIF) | (0 << WDIE) | (1 << WDP3) | (0 << WDCE) | (1 << WDE) | (0 << WDP2) | (0 << WDP1) | (1 << WDP0);
```

فعال کردن اینتراپتهای میکرو:

```
	sei();
}
```

تابع ارسال یک بایت از طریق یوارت که از دیتاشیت میکرو کپی شده:

```
void USART_Tx(unsigned  char data) {
	while (!(UCSR0A & (1 << UDRE0)));   // Wait for empty transmit buffer
	UDR0 = data;                        // Put data into buffer, sends the data
}
```

تابع ارسال یک رشته از طریق یوارت که از تابع قبلی استفاده میکند:

```
void USART_TxStr(unsigned char *data) {
	while (*data != '\0')
		USART_Tx(*data++);
}
```

تابع ارسال یک رشته ثابت ذخیره شده درفلش میکرو از طریق یوارت:

```
void USART_TxStr_P(const char *data) {
	while (pgm_read_byte(data) != 0x00)
		USART_Tx(pgm_read_byte(data++));
}
```

تابع پاک کردن بافر دریافتی یوارت:

برای دریافت درست پاسخ ماژول، میبایست پاسخ قبلی که در بافر ذخیره شده را پاک کرد.

```
void flushBuff(void) {
	rxIndx = 0;
	for (unsigned char i = 0; i<255; i++)
		rxBuff[i] = 0;
}
```

تابع ریست ماژول که در ان هم پایه متصل به ریست ماژول صفر میشود و هم بصورت نرم افزاری دستور ریست به ماژول ارسال میگردد و در پایان نیز میکرو ریست میشود:

```
void rstA6C(void) {
	DDRC |= 0x02; // A6C's rst pin
	USART_TxStr_P(RST);  _delay_ms(100);
	USART_TxStr_P(RST);  _delay_ms(100);
	DDRC &= 0xFD;
	rstMicro();
}
```

تابع ارسال دستورات AT Command به ماژول که سه پارامتر ورودی دارد:

1. cmd دستوری که میبایست به ماژول ارسال شود
2. res پاسخی مورد انتظار
3. timeout مدت زمانی که درصورت عدم دریافت پاسخ از ماژول، تابع مقدار 1 را بمعنی عدم موفقیت در دریافت پاسخ مناسب برمیگرداند.

این تابع حدود هر 1 ثانیه دستور را برای ماژول ارسال کرده و در صورتیکه از ماژول پاسخ مورد انتظار را دریافت کند بلافاصله مقدار 0 را برمیگرداند وگرنه تا پایان تایم اوت هر ثانیه دستور را ارسال کرده و در صورت عدم دریافت پاسخ مورد نظر در این مدت، تابع مقدار 1 را برمیگرداند.

```
unsigned char a6c(const char *cmd, const char *res, unsigned char timeout) {
	unsigned char r = 1;		// مقداری که تابع برمیگرداند
```

```
	flushBuff();			// پاک کردن بافر ورودی
	rxEnable();			// فعال کردن دریافت یورات
	for (unsigned char retryCntr = 0; retryCntr < timeout;) {  // متغیر شمارنده تلاشها
		asm volatile("wdr");			// ریست تایمر واچداگ
		if(TIFR1&0x01) {			// درصورت سرریز تایمر1، شمارنده افزایش میابد
			USART_TxStr_P(cmd);// ارسال دستور به ماژول
			TIFR1 |= 0x01; // ریست کردن فلگ سرریز تایمر1
			retryCntr++;			// افزودن شمارنده تلاش دوباره
		}
		if (strstr_P(rxBuff, res)) r = 0;	// درصورت دریافت پاسخ موردنظر، مقدار 0
	}
	rxDisable();			// غیرفعال کردن دریافت یورات
	return r;// درصورت عدم دریافت پاسخ مورد نظر مقدار 1 برگردانده میشود
}
```

تابع بستن کانکشنهای باز:

از انجا که ممکن است هنوز کانکشن بازی به سرور از طریق ماژول وجود داشته باشد، جهت اطمینان از عملکرد درست میبایست دستور بستن کانشکن ارسال شود. این تابع پس از ارسال این دستور، در یک بازه زمانی حدود 10 ثانیه ای پاسخ دریافتی از ماژول را چک میکند. پس از دریافت این دستور توسط ماژول، اگر کانکشن بازی موجود باشد بسته شده و اکی و در غیر اینصورت ارور بمیکرو برگردانده میشود.

```
void closeConnections(void){
	unsigned char retryCntr = 0;
	flushBuff();
	USART_TxStr_P(CLOSE);		// ارسال فرمان بستن کانکشنها به ماژول
	rxEnable();
	while (!(strstr_P(rxBuff, OK) || strstr_P(rxBuff, ERROR)) && retryCntr < 10) {
		asm volatile("wdr");
		if(TIFR1&0x01) {
			TIFR1 |= 0x01;
			retryCntr++;
		}
	}
	rxDisable();
}
```

تابع روتین اینتراپت دریافت بایت از یورات که بایتهای دریافتی را درصورت دریافت صحیح و سالم، جهت پردازش توسط برنامه اصلی، درون بافر ذخیره میکند:

```
ISR(USART_RX_vect) {
        unsigned  char udr = UDR0;
        if (!(UCSR0A & ((1<<FE0) | (1<<UPE0) | (1<<DOR0))))
                rxBuff[rxIndx++]  = udr;
}
```

تابع روتین اینتراپت پایه متصل به سوییچ زنگ که درصورت فشرده شدن متغیر مربوطه را تغییر میدهد:

```
ISR(PCINT0_vect) {
        if(!(PINB & (1 << 2))) {
                pb2 = 0;
        }
}
```

تابع اصلی برنامه بصورت زیر میباشد:

```
int main(void) {
```

متغیرهای بکار رفته عبارتند از فلگ عکسبرداری و شمارنده تلاش:

```
unsigned  char takePic = 1, retryCntr = 0;
```

اگر فلگ عکسبرداری برابر با 1 باشد، عملیات عکسبرداری و ارسال عکس به سرور انجام میشود. این فلگ در دو حالت فعال میگردد:

1. فشرده شدن شاسی زنگ
2. دریافت فرمان از سمت سرور

```
init();     // فراخانی تابع راه اندازی اولیه
```

پس از راه اندازی واحدهای میکرو، نوبت کانفیگ اولیه ماژول میباشد. بدین منظور تا دریافت پاسخ اکی از ماژول، میکرو هر ثانیه کامند AT را ارسال میکند. توسط این دستور، ماژول میفهمد که بادریت ارتباطی 9600 بوده و خود را تطبیق میدهد و اکی برمیگرداند. درصورتیکه در بازه زمانی 30 ثانیه ای میکرو پاسخ اکی را از ماژول دریافت نکند، ماژول و خودش را ریست میکند.

```
        if (a6c(AT, OK, 30))
                rstA6C();
```

پس از دریافت اکی از ماژول، دستور خاموش کردن اکو ارسال میگردد تا دستورات ارسالی به ماژول اکو نشود:

```
        if (a6c(ATE0, OK, 5))
                rstA6C();
```

با دریافت موفقیت امیز پاسخ اکی از ماژول، میکرو با ارسال دستور مربوط به رجیستر شدن شبکه و پردازش پاسخ دریافتی از ماژول، منتظر اتصال ماژول به شبکه GSM میماند. درصورتیکه در بازه زمانی 30ثانیه ای ارتباط برقرار نشود، میکرو ماژول و خودش را ریست میکن. درصورت برقراری ارتباط، ماژول پاسخ CREG: 1 را برمیگرداند که بسته به انتن دهی منطقه ممکن است بین 5 تا 30 ثانیه زمان ببرد.

```
if (a6c(CREG, PSTR("+CREG: 1"), 30))
        rstMicro();
```

اجرای کامند مربوط به خاموش کردن پیامهای پیامهای اپراترها و انتظاربرای دریافت پاسخ اکی از ماژول:

```
if (a6c(CNMI, OK, 5))          rstA6C();
```

اجرای کامند فعال کردن فلش زدن ماژول هنگام عکسبرداری و انتظار برای دریافت اکی در پاسخ ماژول:

```
if (a6c(CAMCFG, OK, 5))
        rstMicro();
```

اجرای کامند استارت دوربین و انتظار برای دریافت خط جدید در پاسخ ماژول:

```
if (a6c(CAMSTART, CRLF, 10))
        rstMicro();
_delay_ms(10);  // تاخیر مورد نیاز جهت چشمپوشی از مابقی پاسخ دریافت شده از ماژول
```

ارسال دستور تنظیم پرتکل و اکسس‌پوینت و انتظار 5 ثانیه ای برای دریافت اکی:

```
if (a6c(CGDCONT, OK, 5))
        rstMicro();
```

ارسال دستور تنظیم نام اکسس‌پوینت و یوزر و پسورد و انتظار 5 ثانیه ای برای دریافت اکی:

```
if (a6c(CSTT, OK, 5))
        rstMicro();
```

ارسال دستور اتصال به GPRS جهت دسترسی به اینترنت و انتظار 60 ثانیه ای برای دریافت خط جدید از ماژول:

بسته به کیفیت سیگنال دهی این زمان کاهش میابد.

```
if (a6c(CIICR, CRLF, 60))
        rstMicro();
_delay_ms(100);          // تاخیر مورد نیاز جهت چشمپوشی از مابقی پاسخ دریافت شده از ماژول
```

پس از اجرای دستورات بالا اگر کانکشن GPRS بدون مشکل برقرار شود، ماژول ادرس IP دریافت کرده و توسط کد زیر میتوان از دریافت ادرس IP مطمین شد وگرنه سیستم را ریست نمود:

با چک کردن وجود کرکتر نقطه(".") در پاسخ دریافتی از ماژول میتوان از دریافت صحیح IP ادرس اطمینان حاصل کرد.

```
USART_TxStr_P(CIFSR);
flushBuff();
rxEnable();
```

```
while (!strstr(rxBuff, ".") && retryCntr < 10) {
        asm volatile("wdr");
        if(TIFR1&0x01) {
                TIFR1 |= 0x01;
                retryCntr++;
        }
}
rxDisable();
if(retryCntr >= 10)
        rstA6C();
_delay_ms(10);
```

پس از صدور دستورات بالا و اطمینان از اتصال اینترنت، حلقه اصلی برنامه اجرا میشود که در ابتدای ان فلگ عکسبرداری چک میشود و درصورت ست بودن ان، دستور کپچر تصویر و ارسال به سرور صادرمیگردد.

```
for(;;){
        if(takePic) {
```

ارسال فرمان کپچر عکس که باید در بازه زمانی کمتر از 5 ثانیه پاسخ ماژول دریافت شود وگرنه سیستم ریست میشود:

```
                USART_TxStr_P(ATCAMCAP);
                flushBuff();
                rxEnable();
                while (!strstr_P(rxBuff, CAMCAP) && retryCntr < 5) {
                        asm volatile("wdr");
                        if(TIFR1&0x01) {
                                TIFR1 |= 0x01;
                                retryCntr++;
                        }
                }
                rxDisable();
                if(strstr_P(rxBuff, ERROR) || (retryCntr >= 5) )
                        rstA6C();// درصورتیکه پاسخ مناسب دریافت نشود سیستم ریست میگردد
                _delay_ms(100);// تاخیر مورد نیاز جهت نادیده گرفتن باقی پاسخ ماژول
```

بستن کانکشنهای باز جهت اطمینان:

```
closeConnections();
```

اجرای دستور ارسال عکس به سرور:

این مرحله بسته به کیفیت کانکشن ممکن است بین 10 تا 60 ثانیه زمان ببرد که درصورت عدم دریافت پاسخ در این بازه زمانی سیستم ریست میگردد.

```
USART_TxStr_P(ATCAMPOST);
flushBuff();
rxEnable();
retryCntr = 0;
while (!(strstr_P(rxBuff, NORESPONSE) || strstr_P(rxBuff, ERROR) || strstr_P(rxBuff, CAMPOST))
&& retryCntr < 60) {
        asm volatile("wdr");
        if(TIFR1&0x01) {
                TIFR1 |= 0x01;
                retryCntr++;
        }
}
_delay_ms(100);
rxDisable();
```

درصورت عدم موفقیت در ارسال تصویر به سرور، ماژول با 0:CAMPOST پاسخ میدهد که باعث ریست سیستم میشود:

```
if(strstr_P(rxBuff, CAMPOST0) || (retryCntr >= 60) )
        rstA6C();
```

درصورت موفقیت ماژول در ارسال تصویر به سرور، فلگ مربوطه ریست میشود:

```
if(strstr_P(rxBuff, CAMPOST1))          takePic = 0;
}       // اتمام بلاک مربوط به چک کردن فلگ ارسال تصویر
```

در روتین اصلی برنامه وقتی فلگ عکسبرداری ست نشده باشد، اتصال به وبسرور جهت ارسال وضیت ورودی شاسی زنگ و دریافت فلگهای مربوط به خروجیها برقرار میگردد. بدین منظور میکرو دستور مربوطه به ماژول ارسال کرده و تا 10 ثانیه منتظر برقراری ارتباط میماند که درصورت عدم موفقیت سیستم ریست میگردد:

```
retryCntr = 0;
flushBuff();
USART_TxStr_P(WEB); // اجرای دستور برقراری سوکت تی سی پی به پرت 80 سرور اسمارت-دیوایس
rxEnable();
while (!(strstr_P(rxBuff, OK) || strstr_P(rxBuff, ERROR)) && retryCntr < 10) {
        asm volatile("wdr");
```

```
            if(TIFR1&0x01) {
                TIFR1 |= 0x01;
                retryCntr++;
            }
        }
        if (strstr_P(rxBuff, OK)) {
            retryCntr = 0;
        }
        rxDisable();
        if (retryCntr >= 10) {
            rstA6C();// پس از 10 ثانیه اگر ارتباط با وبسرور برقرار نشود سیستم ریست میشود
```

درصورت موفق بودن برقراری ارتباط با سرور، کدهای زیر اجرا میگردد:

```
        } else {
            flushBuff();
            rxEnable();
            USART_TxStr_P(SEND); // اجرای دستور شروع ارسال دیتا
```

درصورت موفقیت امیز بودن برقراری سوکت TCP به پرت 80(پرت مربوط به سرور HTTP)، ماژول با ارسال کرکتر > به میکرو امادگی کانکشن را اعلام میکند و میکرو هم هدر متد GET را ارسال میکند.

```
            for (retryCntr = 0; retryCntr < 10;) {
                if (strstr(rxBuff, ">")) { // درصورت موفقیت ماژول کرکتر > را برمیگرداند
                    USART_TxStr_P(GET); // ارسال هدر مربوط به متد گت
                    if(pb2) {           // درصورت فشرده نشدن ورودی شاسی زنگ، عدد 3 ارسال میشود
                        USART_Tx('3');          // عدد 3 تاثیری در سرور نداشته و نادیده گرفته میشود
                    } else {            // درصورت فشرده شدن زنگ، عدد 1 ارسال میشود
                        USART_Tx('1');
                        takePic = 1;        // فعال کردن فلگ عکسبرداری
                        pb2 = 1;            // غیرفعال کردن فلگ شاسی زنگ
                    }
                    USART_TxStr_P(HTTP);        // ارسال ورژن پرتکل هایپرتکست
                    USART_TxStr_P(HOST);        // ارسال نام سرور که همان اسمارت-دیوایس میباشد
                    USART_Tx(0x1A);// کد اسکی 26 جهت اطلاع ماژول از اتمام دیتا ارسال میگردد
```

```
USART_TxStr_P(CRLF);          // در پایان هم کرکترهای خط جدید ارسال میگردد
```

پس از ارسال درخواست GET، سرور فلگ مربوط به خروجیها را برای ماژول برمیگرداند:

```
flushBuff();
rxEnable();
for (retryCntr = 0; retryCntr < 10;) {
        if (strstr_P(rxBuff, CODE1)) {
```

اگر بافر شامل 1CODE یا همان رشته "1UUU" باشد، LED قرمزغیرفعال شده و سبز فعال میگردد.

```
                PORTD &= 0x7F;          _delay_ms(50);  PORTB |= 0x01;
                break;
        } else if (strstr_P(rxBuff, CODE2)) {
```

اگر بافر شامل 2CODE یا همان رشته "2UUU" باشد، LED سبز غیرفعال شده و قرمز فعال میگردد.

```
                PORTB &= 0xFE;          _delay_ms(50);  PORTD |= 0x80;
                break;
        } else if (strstr_P(rxBuff, CODE3)) {
```

اگر بافر شامل 3CODE یا همان رشته "3UUU" باشد، کاربر درخاست عکسبرداری ارسال کرده و فلگ مربوطه فعال میگردد.

```
                takePic = 1;
                break;
        }
        asm volatile("wdr");
        if(TIFR1&0x01)  {
                TIFR1 |= 0x01;
                retryCntr++;
        }
}
rxDisable();
if(retryCntr<10)
        break;
}
asm volatile("wdr");
if(TIFR1&0x01)  {
        TIFR1 |= 0x01;
```

```
                    retryCntr++;
                }
            }
            rxDisable();

            closeConnections();
        }       // پایان ارتباط با وب سرور
    }           // پایان حلقه اصلی برنامه
}               // پایان تابع اصلی برنامه
```

جزییات پرژه: نرم افزار

از انجا که IP ادرس اختصاص یافته به ماژول در ارتباط GPRS از نوع داینامیک Invalid میباشد، نمیتوان از طریق اینترنت مستقیما با ان ارتباط برقرارنمود. بنابراین نیاز به یک سرور واسطه میباشد تا دیتا را بین دستگاه و کاربر منتقل کند. از انجا که ماژول A6C میتواند تصاویر را به روش POST ارسال نماید، تنها به یک وب سرور یا هاست نیاز خواهیم داشت که بتواند درخاستهای GET و POST را هندل نماید. هاست مورد استفاده در این پرژه از زبان اسکریپتینگ PHP پشتیبانی میکند. PHP یک زبان اسکریپتینگ است که سمت سرور اجرا شده و میتوان خروجی ان را توسط کدینگ HTML روی یک مرورگر وب مشاهده کرد. ساختار فلدر بصورت زیر میباشد که در ادامه توضیح داده خاهد شد:

```
http://smart-device.ir/ic/                  فلدر برنامه سمت سرور
    |   accept.png                          ایکن کلید اکسپت
    |   bell.gif                            ایکن زنگ
    |   bell.png                            جایگزین ایکن زنگ
    |   cntr.txt                            شمارنده تعداد تصاویر ذخیره شده
    |   index.html                          اینترفیس برنامه تحت وب که کاربر مشاهده میکند
    |   key.txt                             متغیر وضعیت شاسی زنگ
    |   mhsn.php                            فایلی که دریافت/ارسال فلگها را انجام میدهد
    |   notif.mp3                           صدای زنگ
    |   reject.png                          ایکن ریجکت
    |   rel.txt                             متغیر رله خروجی
    |   up.php                              فایلی که تصاویر را دریافت و در سرور ذخیره میکند
    └───pics                                فلدری که تصاویر در ان ذخیره میشود
            0.jpg                           تصویر دریافتی
            1.jpg                           1تصویر ذخیره شده شماره
            2.jpg                           2تصویر ذخیره شده شماره
```

3تصویر ذخیره شده شماره	3.jpg
…	
7تصویر ذخیره شده شماره	7.jpg
8تصویر ذخیره شده شماره	8.jpg
9تصویر ذخیره شده شماره	9.jpg
اینترفیسی که تصاویر ذخیره شده را نمایش میدهد	Index.php

روش کار بدین صورت است که دستگاه و اینترفیس وب داده ها را توسط درخاست GET به فایل mhsn.php فرستاده و مقادیر متغیرهای مورد نیاز که در فایلهای تکست ذخیره شده اند را از ان دریافت میکنند.

توضیح اینترفیس وب **index.html**

این فایل شامل کدینگ HTML جهت نمایش رابط کاربری بوده و با استفاده از JavaScript که یک زبان اسکریپتینگ سمت کاربر میباشد و همچنین بهره گیری از تکنیک AJAX، مقادیر ذخیره شده را از فایلهای تکست دریافت کرده و کلیکهای کاربر را از طریق فایل mhsn.php برای ارسال به دستگاه ذخیره مینماید.

```
<html>                                                          شروع فایل
<head>                                                          بخش هد فایل
    <meta http-equiv="Content-Type" content="text/html; charset=us-ascii">      مشخصات محتوای فایل
    <title>Intercom</title>                                     عنوان فایل
    <meta name="viewport" content="width=device-width, initial-scale=1" />      نحوه نمایش فایل در دستگاه
```

جهت نمایش مناسب در گوشی، در اینترفیس وب از زبان استایل شیت CSS و کتابخانه اماده بوت استرپ استفاده شده است. به کمک این کتابخانه، تصویر دریافتی بصورت تمام-عرض در دستگاه نمایش نشان داده شده و متناسب با سایز گوشی تغییر میکند. توضیحات کامل درباره این کتابخانه از سایت بوت استرپ به ادرس https://getbootstrap.com/getting-started/ در دسترس میباشد.

```
<link href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css" rel="stylesheet" />
```

در بخش استایل، ویژگیهای نمایشی یک اسنک بار مشخص شده. اسنک بار هنگام کلیک روی کلیدهای برنامه بصورت یک اعلان در پایین صفحه نمایش داده میشود. توضیحات بیشتر در این لینک: https://www.w3schools.com/howto/howto_js_snackbar.asp

```
<style type="text/css">
#snackbar {
    visibility: hidden;
    margin-left: -50px;
    background-color: #333;
    color: #fff;
    text-align: center;
    border-radius: 20px;
    padding: 16px;
```

```
    position: fixed;
    z-index: 1;
    left: 50%;
    bottom: 30px;
    font-size: 17px;
}

#snackbar.show {
    visibility: visible;
    -webkit-animation: fadein 0.5s, fadeout 0.5s 2.5s;
    animation: fadein 0.5s, fadeout 0.5s 2.5s;
}

@-webkit-keyframes fadein {
    from {bottom: 0; opacity: 0;}
    to {bottom: 30px; opacity: 1;}
}

@keyframes fadein {
    from {bottom: 0; opacity: 0;}
    to {bottom: 30px; opacity: 1;}
}

@-webkit-keyframes fadeout {
    from {bottom: 30px; opacity: 1;}
    to {bottom: 0; opacity: 0;}
}

@keyframes fadeout {
    from {bottom: 30px; opacity: 1;}
    to {bottom: 0; opacity: 0;}
```

```
        }

        img{
            padding: 5px;
        }
        </style>                                                    پایان بخش استایل
</head>                                                             پایان بخش هد فایل
<body style="background-color:#F6FCFF">                            شروع بدنه فایل با پس زمینه ابی کم رنگ
<center>                                                            کل محتویات بدنه وسط چین میشوند
<img class="img-responsive" id="cam" onclick="fcnPic()" src="./pics/0.jpg" />   نمایش تصویر دریافتی
<div class="row">                                                   تعریف یک سطر جهت نمایش ایکنهای اکسپت و ریجکت و زنگ
<div class="col-xs-4"><img class="img-responsive" id="imgAccept" onclick="fcnAccept()" src="./accept.png" /></div>
                                                                    نمایش ایکن اکسپت
<div class="col-xs-4"><img class="img-responsive" id="imgBell" onclick="fcnDismiss()" src="./bell.png" /></div>
                                                                    نمایش ایکن زنگ
<div class="col-xs-4"><img class="img-responsive" id="imgReject" onclick="fcnReject()" src="./reject.png" /></div>
                                                                    نمایش ایکن ریجکت
</div>                                                              پایان سطر مربوط به ایکنها
<div class="jumbotron text-center">
<h3><a href="./pics">Saved Pictures</a></h3>                        نمایش لینک تصاویر ذخیره شده
</div>
<div id="snackbar">Config Sent</div>                                بخش مربوط به اسنک بار
<audio id="sndNotif" preload="auto" src="notif.mp3"></audio>        صدای زنگ
<script>                                                            بخش مربوط به جاواسکریپت که وظیفه بروزرسانی اینترفیس و انتقال فلگها را برعهده دارد
var myVar = setInterval(myTimer, 3000);                             تایمری که هر 3 ثانیه تابع بروزرسانی صفحه را اجرا مینماید
function showSnackbar() {       تابع نمایش اسنک بار که با کلیک کاربر روی ایکنها اجرا شده و ویبره را هم فعال مینماید
  navigator.vibrate = navigator.vibrate || navigator.webkitVibrate || navigator.mozVibrate || navigator.msVibrate;
                                                                    دریفات وبکیت ویبره دستگاه
  if (navigator.vibrate) {
    navigator.vibrate(100);                                         درصورت پشتیبانی دستگاه از ویبره، بمدت 100 میلی ثانیه دستگاه ویبره میکند
  }
  var x = document.getElementById("snackbar")
```

```
    x.className = "show";                                                    نمایش اسنک بار
    setTimeout(function(){ x.className = x.className.replace("show", ""); }, 500);    بمدت 500 میلی ثانیه
}
function fcnAccept() {                    این تابع با کلیک ایکن اکسپت اجرا شده و مقدار 1 را برای ذخیره شدن در متغیر خروجی، ارسال مینماید
    var client = new XMLHttpRequest();                                       متغیر مورد نیاز برای ارسال دیتا
    client.open("GET", "http://smart-device.ir/ic/mhsn.php?o=1");  1 بازکردن درخاست مورد نیاز به روش گت جهت ارسال مقدار
    client.send();                                                           ارسال دیتا که باعث فعال شدن خروجی اکسپت(سبز) میشود
    showSnackbar();                                                          نمایش اسنک بار و ویبره
}
function fcnReject() {                    این تابع با کلیک ایکن اکسپت اجرا شده و مقدار2 را برای ذخیره شدن در متغیر خروجی، ارسال مینماید
    var client = new XMLHttpRequest();                                       متغیر مورد نیاز برای ارسال دیتا
    client.open("GET", "http://smart-device.ir/ic/mhsn.php?o=2");  2 بازکردن درخاست مورد نیاز به روش گت جهت ارسال مقدار
    client.send();                                                           ارسال دیتا که باعث فعال شدن خروجی ریجکت(قرمز) میشود
    showSnackbar();                                                          نمایش اسنک بار و ویبره
}

function fcnPic(){                        این تابع با کلیک روی تصویر دریافتی اجرا شده و مقدار2 را برای ذخیره شدن در متغیر خروجی، ارسال مینماید
    var client = new XMLHttpRequest();                                       متغیر مورد نیاز برای ارسال دیتا
    client.open("GET", "http://smart-device.ir/ic/mhsn.php?o=3");  3 بازکردن درخاست مورد نیاز به روش گت جهت ارسال مقدار
    client.send();                                                           ارسال دیتا که باعث فعال شدن فلگ عکسبرداری میشود
    showSnackbar();                                                          نمایش اسنک بار و ویبره
}
function fcnDismiss(){                    این تابع با کلیک روی ایکن زنگ اجرا شده و مقدار2 را برای ذخیره شدن در متغیر ورودی، ارسال مینماید
    var client = new XMLHttpRequest();                                       متغیر مورد نیاز برای ارسال دیتا
    client.open("GET", "http://smart-device.ir/ic/mhsn.php?i=2");  2 بازکردن درخاست مورد نیاز به روش گت جهت ارسال مقدار
    client.send();                                                           ارسال دیتا که باعث غیرفعال شدن صدای زنگ میشود
    showSnackbar();                                                          نمایش اسنک بار و ویبره
}
function myTimer() {    تایعی که جهت دریافت و نمایش تصویر جدید و وضعیت زنگ و بروزرسانی صفحه بکار میرود و هر 3 ثانیه اجرا میشود
    var d = new Date();    متغیر زمان که جهت متغیر رندم بکار میرود تا تصویری تازه بغیر از تصویر ذخیره شده در کش مرورگر دریافت شود
    document.getElementById("cam").src="./pics/0.jpg?v=" + d.getTime();      بروزرسانی تصویر
```

```
var client = new XMLHttpRequest();                                  متغیر مورد نیاز برای دریافت دیتا
client.open("GET", "http://smart-device.ir/ic/key.txt?v=" + d.getTime());   دریافت متغیر ورودی شاسی زنگ
client.onreadystatechange = function() {                            وقتی دریافت متغیر ورودی شاسی زنگ دریافت شود این تابع اجرا میگردد
if(client.responseText.includes("1")) {                             اگر متغیر ورودی شاسی زنگ شامل عدد 1 باشد یعنی شاسی فشرده شده و زنگ بصدا در می اید
  document.getElementById("imgBell").src="http://smart-device.ir/ic/bell.gif";   نمایش انیمیشن زنگ
  document.getElementById("sndNotif").play();                       پخش صدای زنگ
  navigator.vibrate = navigator.vibrate || navigator.webkitVibrate || navigator.mozVibrate || navigator.msVibrate;
  if (navigator.vibrate) {                                          چک کردن پشتیبانی دستگاه از ویبره
      navigator.vibrate([100,100,100,100,100]);                     فعال شدن ویبره دستگاه در 5 ضرباهنگ 100 میلی ثانیه ای
  }
} else if(client.responseText.includes("2")){                       اگر متغیر ورودی شاسی زنگ شامل عدد 2 باشد یعنی شاسی فشرده نشده
  document.getElementById("imgBell").src="http://smart-device.ir/ic/bell.png";   غیرفعال کردن انیمیشن زنگ
}
}
client.send();                                                      اجرای درخاست دریافت متغیر ورودی شاسی زنگ
}
</script>                                                           پایان جاواسکریپت
</center>                                                           پایان مارکاپ وسط چین
</body>                                                             پایان بدنه فایل
</html>                                                             پایان فایل
```

توضیح فایل **mhsn.php**

این فایل درواقع رابطی بین دستگاه/اپلیکیشن با متغیرهای ذخیره شده در فایلهای تکست میباشد. اگر پارامتر i همراه با مقدار شامل 1 یا 2 یا 3 به این فایل ارسال شود، درون فایل key.txt ذخیره شده و مقدار رله خروجی از فایل rel.txt خانده شده و برای ایفن ارسال میگردد. بدین ترتیب سرور از ورودی شاسی زنگ مطلع شده و برد نیز وضعیتی که باید بر رله خروجی اعمال کند را دریافت مینماید. برای تغییر خروجی رله نیز اپلیکیشن پارامتر o را برای این فایل ارسال مینماید.

شرح پارامترهای ارسالی در جدول زیر توضیح داده شده است:

شرح	پارامتر ارسالی
شاسی زنگ فشرده شده است، ارسالی از طرف برد ایفن	i=1
شاسی زنگ فشرده نشده است، ارسالی از اپلیکیشن جهت سایلنت کردن صدای زنگ	i=2

تغییری در سیستم ایجاد نمیکند و تنها جهت دریافت وضعیت خروجی توسط ایفن ارسال میگردد	i=3
خروجی رله 1(سبز) روشن گردد، ارسالی از طرف اپلیکیشن	o=1
خروجی رله 2(قرمز) روشن گردد، ارسالی از طرف اپلیکیشن	o=2
دستور عکس برداری، ارسالی از طرف اپلیکیشن	o=3

```
<?php                                                                   شروع اسکریپت
$input = $_GET['i'];                                                    دریافت پارامتر ورودی از درخاست گت
    if($input==1 || $input==2 || $input==3) {                            اگر مقدار پارامتر ورودی برابربا 1 یا 2 یا 3 باشد این بخش اجرا میشود
  if($input!=3) {                                                       اگر مقدار پارامتر مخالف 3 باشد، مقدار ان در فایل تکست مربوط به ورودی شاسی زنگ ذخیره میشود
      $ourFileHandle  = fopen("key.txt", 'w') or die("can't open file");  باز کردن فایل تکست جهت نوشتن
          fwrite($ourFileHandle, $input);                               نوشتن مقدار پارامتر ورودی در فایل
        fclose($ourFileHandle);                                         بستن فایل
      }
    $ourFileHandle  = fopen("rel.txt", 'r') or die("can't open file");  باز کردن فایل مربوط به خروجی رله جهت خاندن
          $theData = fread($ourFileHandle , 1);                         خاندن بایت اول فایل در متغیر دیتا
          fclose($ourFileHandle);                                       بستن فایل
          echo("UUU");echo($theData);                                   نمایش دیتا با فرمت تعریف شده
    } else {                                                            اگر مقدار پارامتر ورودی برابربا 1 یا 2 یا 3 نباشد، این بخش اجرا میشود
  $output = $_GET['o'];                                                 دریافت مقدار پارامتر مربوط به خروجی رله
  if($output==1 || $output==2 || $output==3){                           اگر مقدار پارامتر خروجی برابربا 1 یا 2 یا 3 باشد، این بخش اجرا میشود
          $ourFileHandle  = fopen("rel.txt", 'w') or die("can't open file");  بازکردن فایل متغیر رله جهت نوشتن
          fwrite($ourFileHandle, $output);                              نوشتن مقدار خروجی در فایل رله
          fclose($ourFileHandle);                                       بستن فایل رله
  }                                                                     پایان بخش مربوط به نوشتن فایل رله
    }
    fclose($ourFileHandle);                                             بستن فایلهای باز
?>                                                                      پایان اسکریپت
```

توضیح فایل **up.php**

این فایل، عکس را از برد ایفن دریافت کرده و در سرور ذخیره مینماید. جهت دسترسی کاربر، از یک کانتر(نگهداری شده در cntr.txt) بهره گرفته شده تا بتوان 9 عکس را ذخیره نمود.

```
<?php                                                                   شروع اسکریپت
```

```
$uploaddir = "pics/";                                   قلدری که عکسها در ان نگهداری میشوند
$uploadfile=$_GET['n'];                                 دریافت نام فایل اپلود شده


if (is_uploaded_file($_FILES['photo']['tmp_name'])) {   اطمینان از صحت اپلود عکس
   if ($_FILES["photo"]["size"] > 500000) {             اگر حجم عکس از 500 کیلو بیشتر باشد نادیده گرفته خاهد شد
      echo " File is too large.";
   } else {                                             درصورتیکه حجم فایل در حد مجاز باشد ذخیره خاهد شد
      move_uploaded_file($_FILES['photo']['tmp_name'], $uploaddir.$uploadfile);انتقال عکس اپلود شده به فلدر نگهداری
      $ourFileHandle = fopen("cntr.txt", 'r') or die("can't open file");  باز کردن فایل کانتر جهت خاندن
      $theData = fread($ourFileHandle , 1);             خاندن 1 بایت اول فایل کانتر و نگهداری در متغیر دیتا
      fclose($ourFileHandle);                           بستن فایل کانت
      if($theData<9) $theData++;                        اگر مقدار کانتر از 9 کمتر باشد یکی به ان افزوده میگردد
      else $theData=1;                                  وگرنه مقدارش 1 میگردد
      $ourFileHandle = fopen("cntr.txt", 'w') or die("can't open file");  بازکردن فایل کانتر جهت نوشتن
      fwrite($ourFileHandle, $theData);                 نوشتن مقدار متغیر دیتا(شامل مقدار جدید کانتر) در فایل کانتر
      fclose($ourFileHandle);                           بستن فایل کانتر
      copy($uploaddir.$uploadfile,$uploaddir.$theData.".jpg");  کپی کردن عکس اپلد شده با نام کانتر
      echo " UpOK";                                     نمایش موفقیت امیز بودن اپلد
   }
} else {                                                درصورت عدم موفقیت در اپلد عکس، پیام خطا نمایش داده میشود
   echo " erRor";
}
?>                                                      پایان اسکریپت
```

توضیح فایل **pics/index.php**

این فایل عکسهای نگهداری شده در فلدر pics را برای کاربر نمایش میدهد که جهت نمایش مناسب در تمامی دستگاهها از جمله گوشیهای مبایل، از زبان استایل شیت CSS بهره گرفته شده است. برای اشنایی با CSS در لینک https://www.w3.org/standards/webdesign/htmlcss توضیحات بسیار خوبی داده شده است.

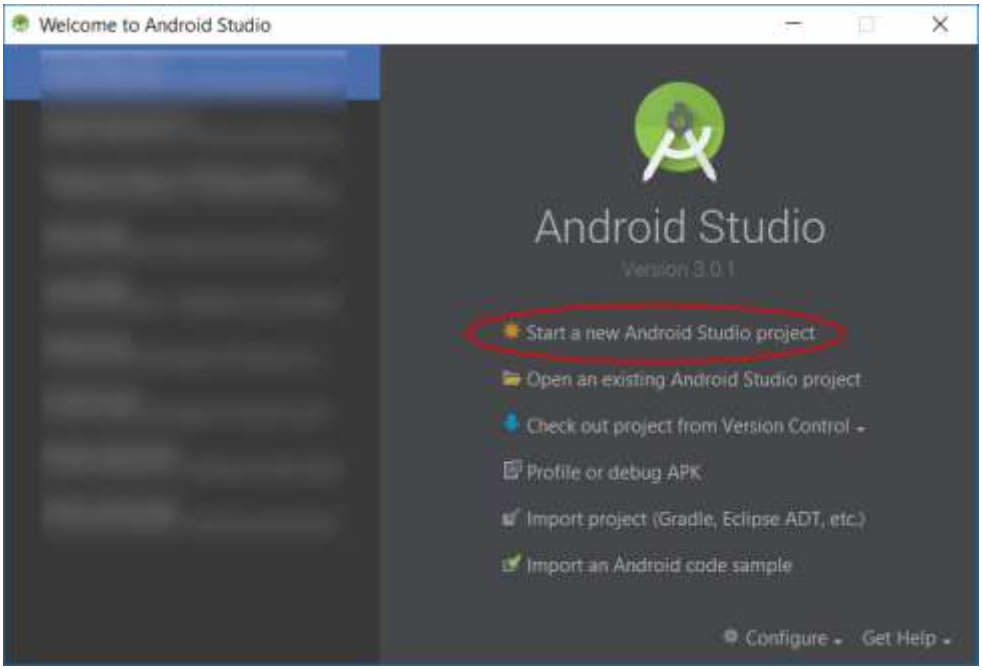
```
<html>                                                  شروع فایل
<head>                                                  بخش هد فایل شامل کدهای استایل شیت
<style>                                                 کدهای استایل شیت
div.gallery {
```

```
    border: 1px solid #ccc;
}

div.gallery:hover {
    border: 1px solid #777;
}

div.gallery img {
    width: 100%;
    height: auto;
}

div.desc {
    padding: 15px;
    text-align: center;
}

* {
    box-sizing: border-box;
}

.responsive {
    padding: 0 6px;
    float: left;
    width: 24.99999%;
}

@media only screen and (max-width: 700px) {
    .responsive {
        width: 49.99999%;
        margin: 6px 0;
```

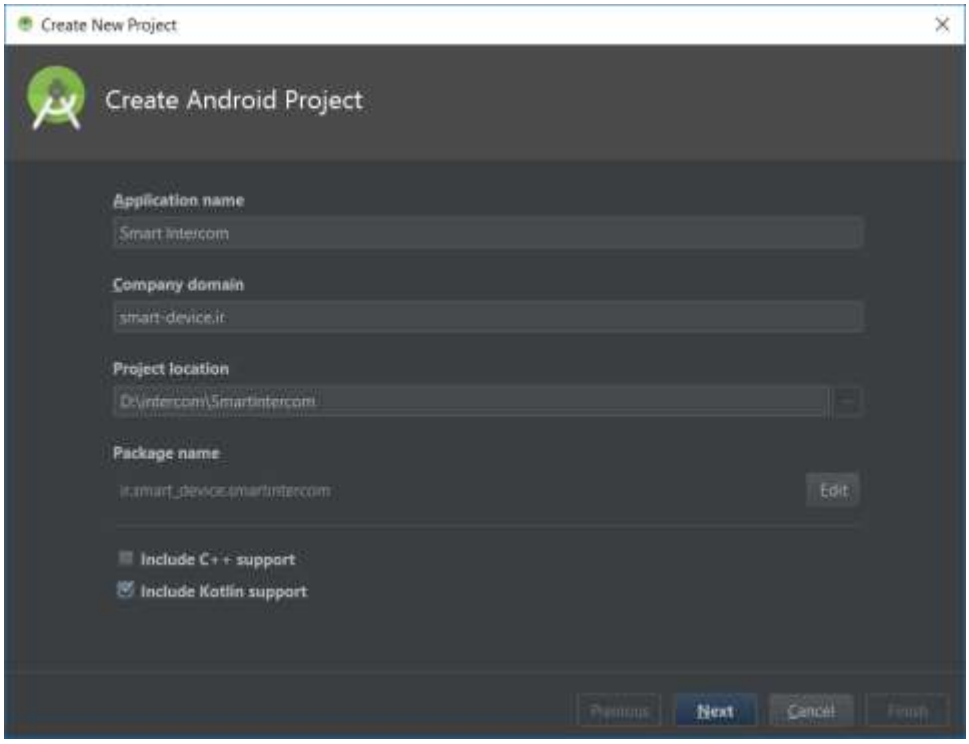
```
    }
}

@media only screen and (max-width: 500px) {
    .responsive {
        width: 100%;
    }
}

.clearfix:after  {
    content: "";
    display: table;
    clear: both;
}
</style>                                                                    پایان کدهای استایل شیت
</head>                                                                     پایان بخش هد فایل
<body>                                                                      بدنه فایل
<p><button onclick="window.history.back();">Go Back</button></p>        کلید برگشت به صفحه قبل
<?php                                                                       جهت نمایش عکس و نام و مشخصاتش نیاز به اسکریپت است که در این بخش نوشته شده
function format_size($size) {                                               تابعی که سایز فایل را بصورت کاربرپسند نمایش میدهد
    $sizes = array(" Bytes", " KB", " MB", " GB", " TB", " PB", " EB", " ZB", " YB");
    if ($size == 0) { return('n/a'); } else {
    return (round($size/pow(1024, ($i = floor(log($size, 1024)))), 2) . $sizes[$i]); }
}
$images = glob("*.jpg");                                                    نگهداری نام عکسهای موجود در این فلدر بصورت ارایه
foreach($images as $image) {                                                برای تمامی عکسها، این حلقه اجرا شده و عکسها بهمراه نام و مشخصاتش نمایش داده میشوند
    echo '<div class="responsive"><div class="gallery"><a href="'.$image.'"><img src="'.$image.'" alt=$image></a><div
class="desc">'.$image.' '.format_size(filesize($image)).'<br />'.date('F d Y H:i:s',filemtime($image)).'</div></div></div>';
}
?>                                                                          پایان اسکریپت
</body>                                                                     پایان بدنه فایل
</html>                                                                     پایان فایل
```

توضیح اپلیکیشن اندرید

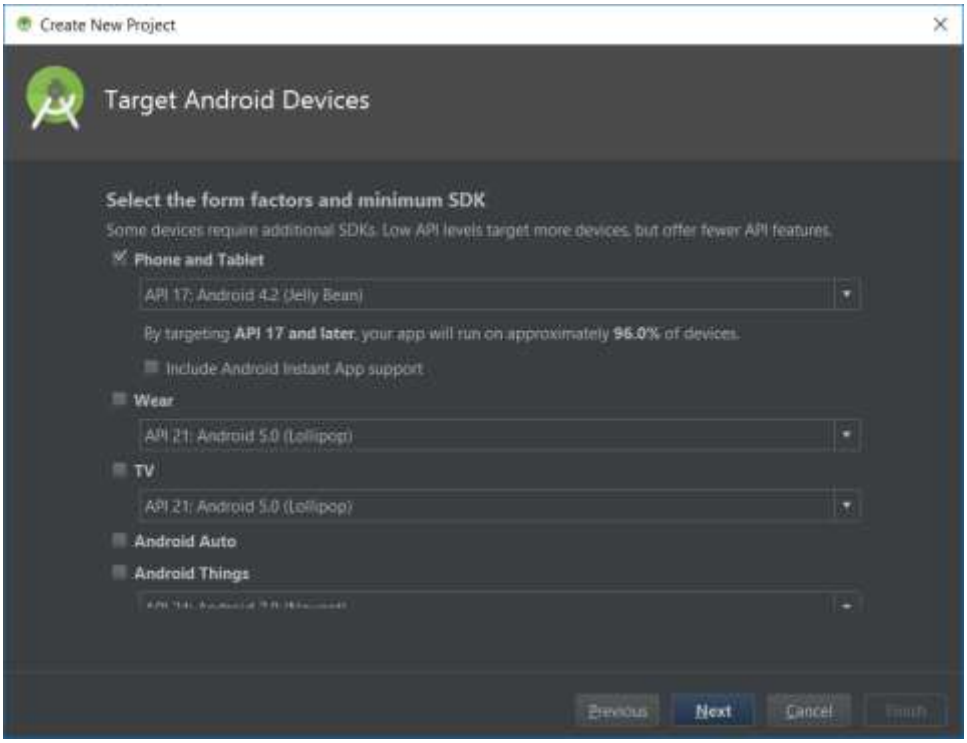
بیشتر کار مربوط به اینترفیس کاربری در بخش اینترفیس وب انجام شده و برای اپ اندریدی، به یک webview و کانفیگ مناسب ان نیاز داریم. بدین منظور از برنامه Android Studio که IDE رسمی شرکت گوگل برای توسعه اپهای اندریدی میباشد بهره گرفته شده. زبان برنامه نویسی مورد استفاده Kotlin است که به تازگی توسط گوگل پشتیبانی و سفارش شده است. از مزیتهای کاتلین نسبت به جاوا میتوان به سادگی ان اشاره نمود. برای برنامه نویسی کاتلین میبایست اخرین ورژن اندرید استادیو را از سایت https://developer.android.com/studio/index.html دانلد و نصب نمود. پس از نصب و اجرای برنامه گزینه استارت یک پرژه جدید را انتخاب میکنیم:



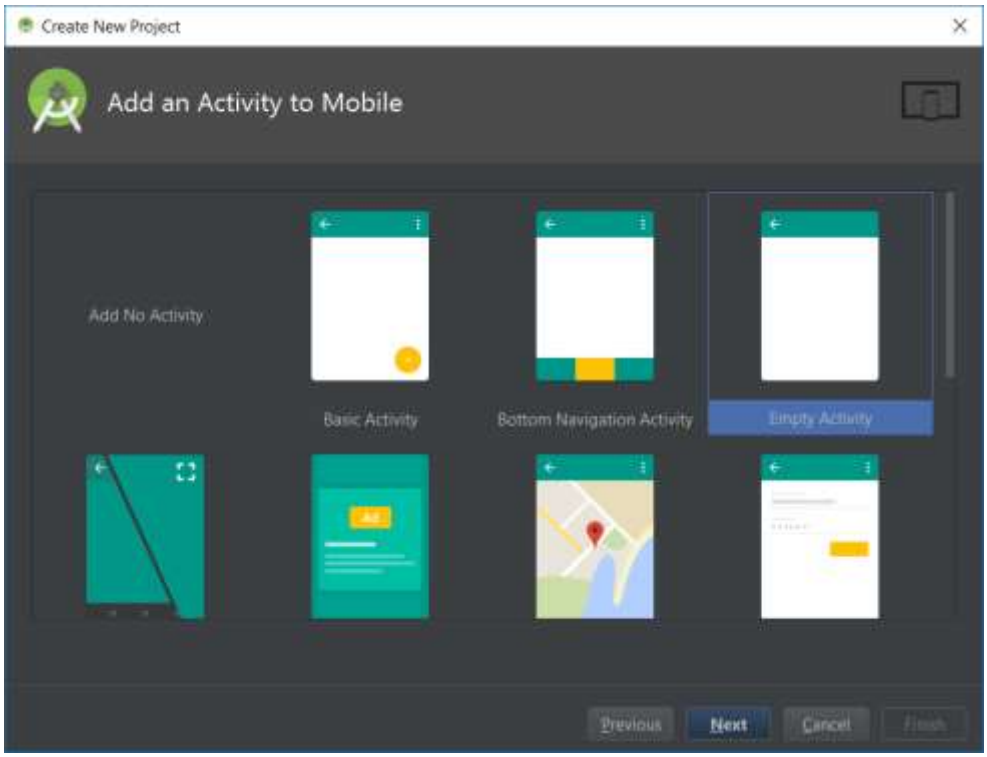
در کادر باز شده نام اپلیکیشن و سایر مشخصات را وارد کرده و گزینه پشتیبانی از کاتلین در پایین کادر را هم فعال میکنیم:



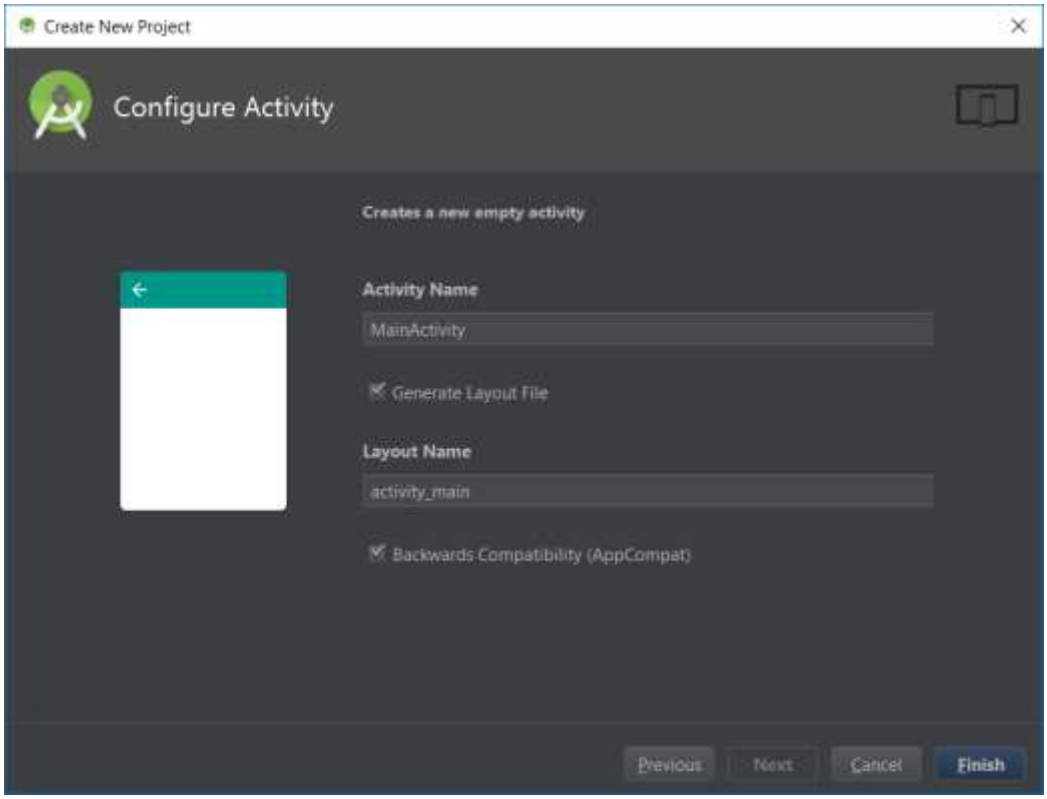
در مرحله بعد دستگاههای هدف را انتخاب میکنیم که در اینجا گوشی و تبلتهای با اندرید 4.2 به بالا مدنظر است:



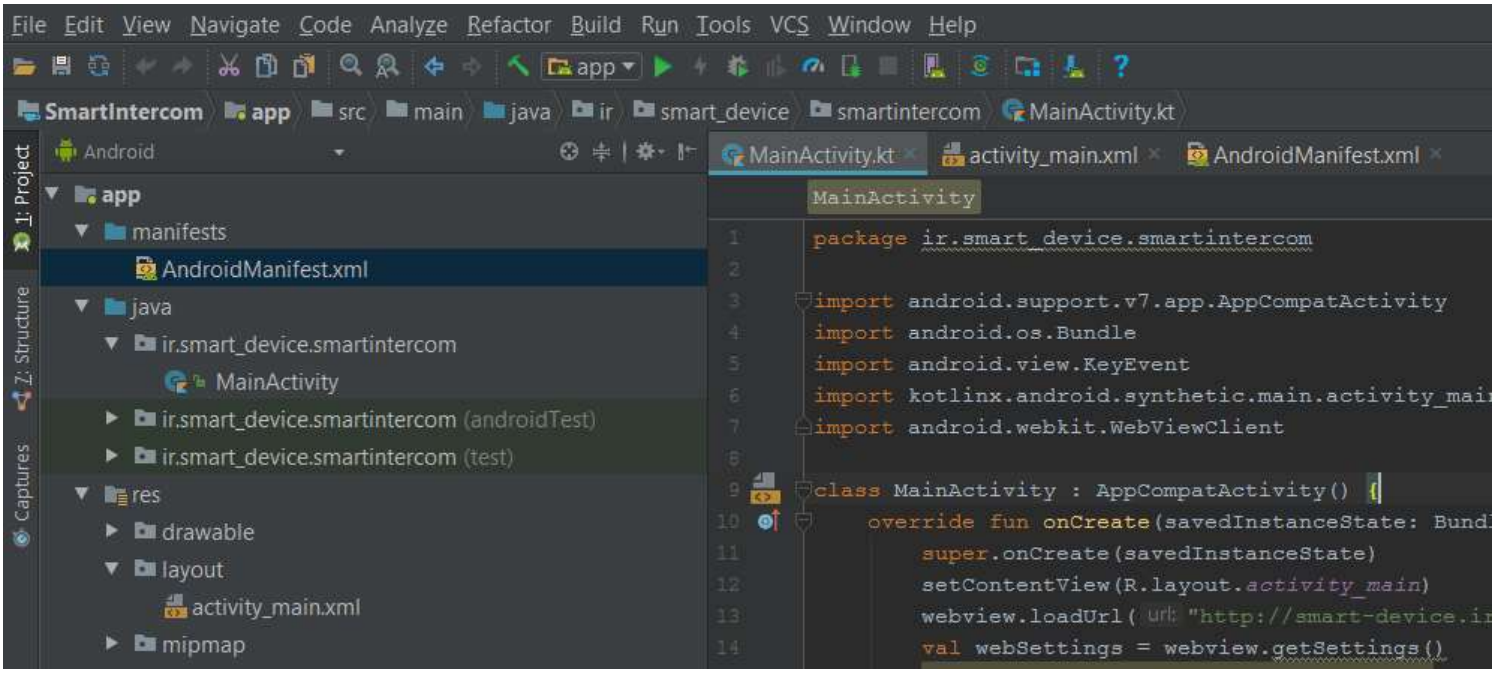
در بخش افزودن اکتیویتی، یک اکتیویتی خالی برگزیده:



و انرا MainActivity مینامیم و روی پایان کلیک میکنیم:



توضیحات کامل درباره برنامه نویسی اندرید در سایت https://developer.android.com/ در دسترس میباشد.



نمایی از پرژه ایجاد شده

هر پرژه شامل یک فایل مانیفست میباشد که مشخصات کلی پرژه در ان تعریف میگردد. جهت دسترسی برنامه به اینترنت و همچنین بهره گیری از ویبره دستگاه، این دو خط را به فایل AndroidManifest.xml میافزاییم:

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.VIBRATE"/>
```

هر اکتیویتی یا صفحه که نقش اینترفیس کاربری را دارد، توسط یک فایل xml تعریف میشود و با یک فایل kt نیز در تعامل است. اکتیویتی این پرژه activity_main.xml تنها شامل یک WebView میباشد:

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="ir.smart_device.smartintercom.MainActivity">

    <WebView  xmlns:android="http://schemas.android.com/apk/res/android"
        android:id="@+id/webview"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        />
</android.support.constraint.ConstraintLayout>
```

فایل MainActivity.kt شامل برنامه اصلی میباشد که در ابتدا با نام پکیج اغاز میگردد:

```
package ir.smart_device.smartintercom
```

ایمپرتهای مورد استفاده در این برنامه:

```
import android.support.v7.app.AppCompatActivity
import android.os.Bundle
import android.view.KeyEvent
import kotlinx.android.synthetic.main.activity_main.*
import android.webkit.WebViewClient
```

کلاس اصلی برنامه

```
class MainActivity : AppCompatActivity() {
```

تابع اصلی برنامه که هنگام ایجاد اکتیویتی اجرا میگردد

```
    override fun onCreate(savedInstanceState: Bundle?) {
```

سوپرکلاس ویرچوال ماشین دالویک

```
        super.onCreate(savedInstanceState)
```

تعیین صفحه اکتیویتی مرتبط با این برنامه

```
        setContentView(R.layout.activity_main)
```

تعیین ادرس وب مورد استفاده در وب ویو

```
        webview.loadUrl("http://smart-device.ir/ic/")
```

متغیری که برای کانفیگ وب ویو بکار میرود

```
        val webSettings = webview.getSettings()
```

فعال کردن جاواسکریپت که بصورت دیفتات غیرفعال میباشد

```
        webSettings.setJavaScriptEnabled(true)
```

فعال کردن امکان پیمایش و باز کردن لینکها در وب ویو

```
        webview.setWebViewClient(WebViewClient())
    }
```

تابعی که هنگام کلیک روی ایکن عقبگرد ران میشود و امکان پیمایش به وب ویو میدهد:

```
    override fun onKeyDown(keyCode: Int, event: KeyEvent): Boolean {
        // Check if the key event was the Back button and if there's history
        if (keyCode == KeyEvent.KEYCODE_BACK && webview.canGoBack()) {
            webview.goBack()
            return true
        }
        // If it wasn't the Back key or there's no web page history, bubble up to the default
        // system behavior (probably exit the activity)
        return super.onKeyDown(keyCode, event)
    }
}
```

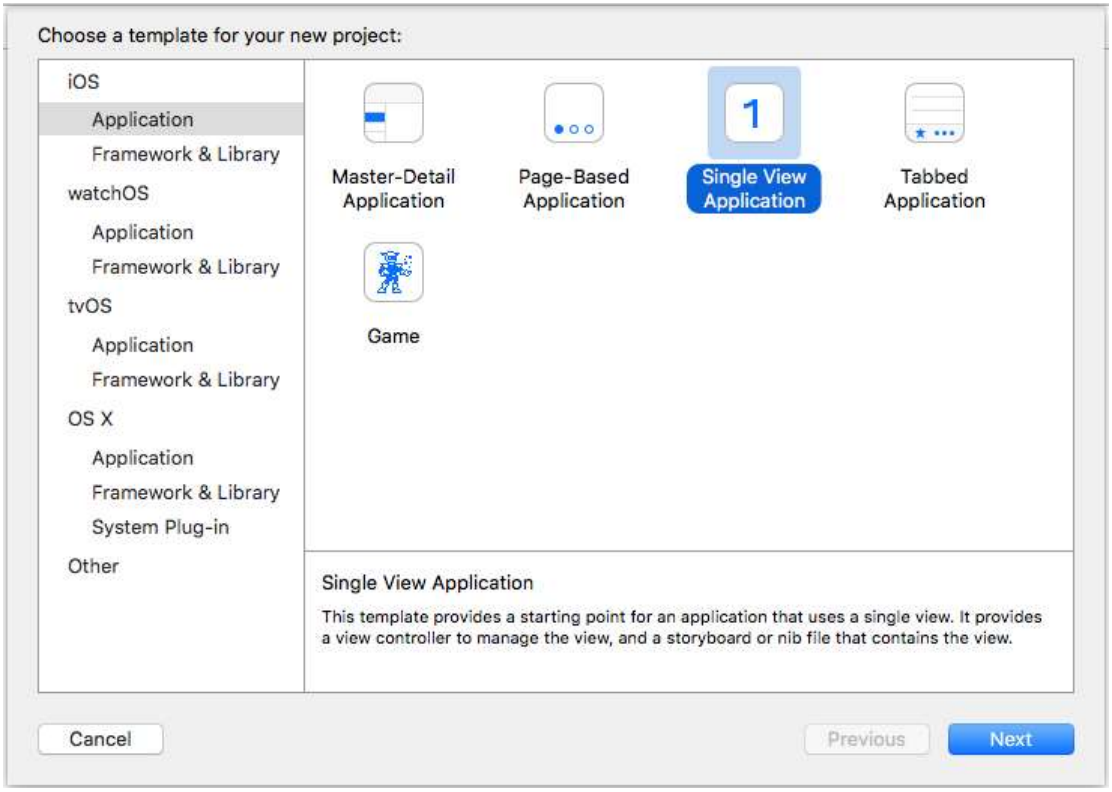
پس از اعمال تغییرات بالا در فایلهای پرژه، میتوان از طریق Build پرژه را کامپایل کرده و فایل APK را جهت اجرا روی گوشی بدست اورد و یا با اتصال گوشی و فعال کردن قابلیت دیباگ از طریق USB، پرژه را ران و دیباگ نمود.

توضیح اپلیکیشن **iOS**

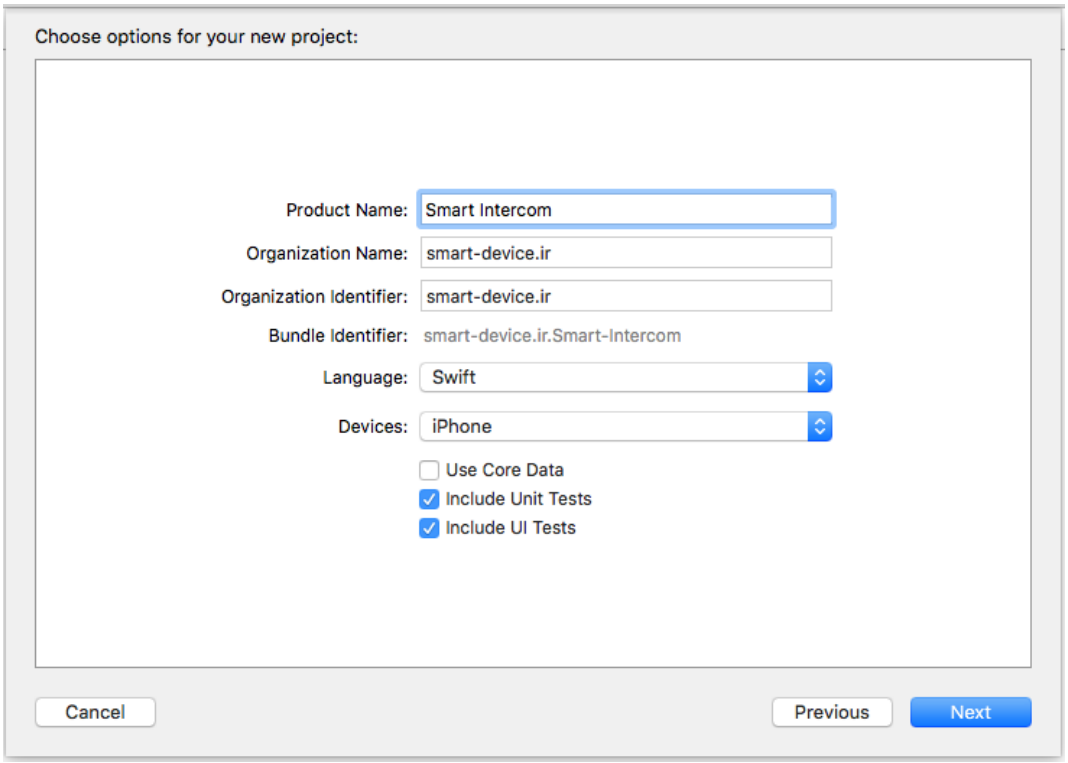
برای ساخت برنامه برای پلتفرم iOS، نیاز به Xcode IDE میباشد که میتوان از لینک https://developer.apple.com/download/ دانلد نمود. راهنما و توضیحات کامل نیز از طریق سایت https://developer.apple.com/documentation/ در دسترس است.

پس از دانلد و نصب Xcode، انرا اجرا کرده و یک پرژه جدید ایجاد میکنیم:

در کادر باز شده اپلیکیشن تک نما را برگزیده و نکست میکنیم:

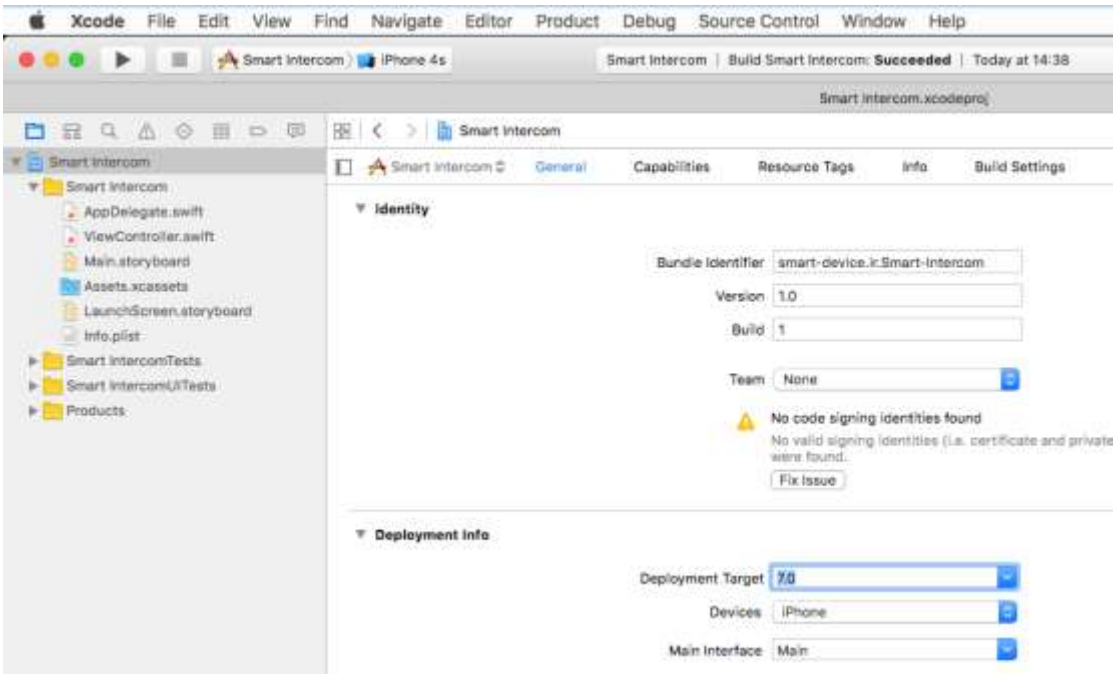


نام پرژه و سایر مشخصات را وارد کرده و زبان برنامه نویسی را سوییفت انتخاب میکنیم. سوییفت جایگزین پیشنهادی اپل برای توسعه اپ میباشد.



در پایان مکان ذخیره پرژه را مشخص کرده و با کلیک روی Create پرژه ایجاد میگردد.

سوییفت روی iOS ورژن 7 به بعد کار میکند. بنابراین میبایست از بخش Deployment Info گزینه Deployment Target روی ورژن 7.0 تنظیم گردد:



هر اینترفیس اپ iOS شامل حداقل یک فایل استوری برد Main.storyboard و یک ویوکنترلر ViewController.swift میباشد. استوری برد اینترفیس گرافیکی است که کاربر میبیند و کنترلر برنامه مرتبط با ان میباشد. در این پرژه برای اضافه نمودن WebView به استوری برد از دستورات سوییفت بهره گرفته شده که در هنگام اجرای برنامه، وب ویو به اینترفیس افزوده میشود. بدین منظور کد زیر میبایست پس از کد مربوط به بارگزاری اینترفیس افزوده گردد:

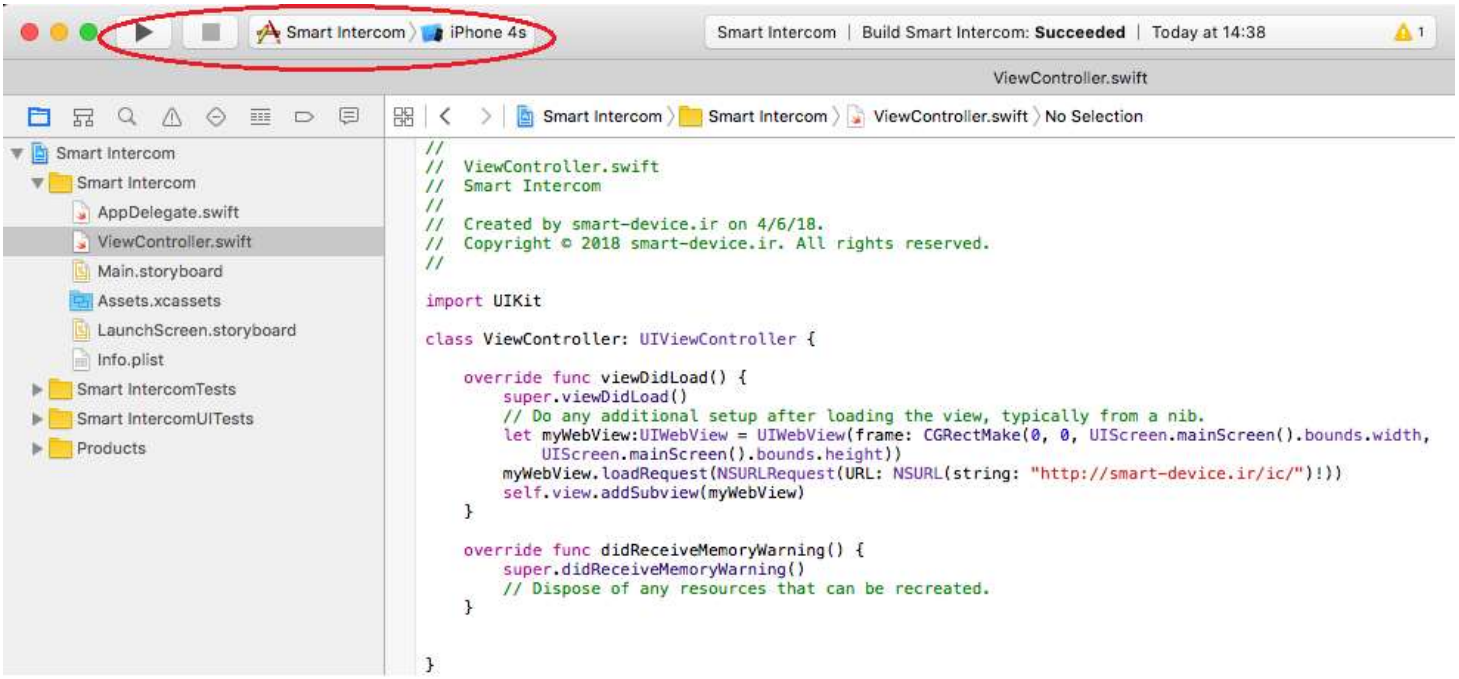
let myWebView:UIWebView = UIWebView(frame: CGRectMake(0, 0, UIScreen.mainScreen().bounds.width, UIScreen.mainScreen().bounds.height)) معرفی وب ویو بعنوان یک ویوی تمام صفحه

myWebView.loadRequest(NSURLRequest(URL: NSURL(string: "http://smart-device.ir/ic/")!)) ادرس اینترنتی وب ویو

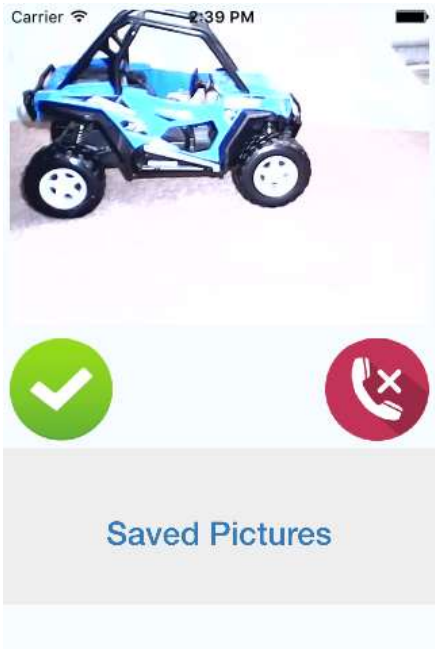
self.view.addSubview(myWebView) افزودن وب ویو به اینترفیس اپ

توضیحات کامل وب ویو در سایت اپل به ادرس https://developer.apple.com/documentation/webkit/webview در دسترس میباشد.

سپس در بخش اجرای اپ در سمت چپ-بالا، دستگاه هدف را تعیین کرده و روی ران کلیک میکنیم:



اپ کامپایل شده و درون شبیه ساز اجرا میگردد:



البته از انجا که دستگاههای اپل فاقد کلید عقبگرد میباشند، نیاز به افزودن این کلید بصورت نرم افزاری میباشد که در اسناد اپل بطور کامل توضیح داده شده است:

https://developer.apple.com/library/content/documentation/Cocoa/Conceptual/DisplayWebContent/Tasks/BackForwardList.html

دانلد سرس کد پرژه:

https://github.com/smart-device/Smart-Intercom/

دیتاشیت ماژول:

https://www.makerfabs.com/desfile/files/A6_A7_A6C_datasheet-EN.pdf

دستورات AT Command ماژول:

https://www.makerfabs.com/desfile/files/A6_A7_A6C_A20%20AT%20Command%20V1.03.pdf

تهیه شده در گروه فنی مهندسی طراحان الکترنیک

97/01/15

http://smart-device.ir

@android_electronic_project

با سپاس از جناب اقای مهندس کی‌نژاد(knowledgeplus.ir)