Sistema IoT para el Monitoreo y Control Eficiente de la Iluminación en Espacios Urbanos de Tijuana

EcoLuz TJ

Caso practico 3

Introducción

- En Tijuana, el uso innecesario de iluminación en espacios públicos y privados representa una oportunidad de ahorro energético que pocas veces se aprovecha.
- Detectar y automatizar el control de la luz puede mejorar significativamente el consumo responsable.
- Este proyecto propone un sistema con ESP32 que mide la luz ambiental con una fotoresistencia, controla un LED en función de esa lectura, y envía los datos a Firebase Realtime Database usando HTTP. Esto permite registrar en la nube los cambios en tiempo real.
- La solución facilita el monitoreo ambiental desde cualquier lugar, fomenta el aprendizaje en tecnologías IoT, y promueve el uso eficiente de la energía. Además, los datos podrán verse en una aplicación móvil creada con App Inventor.

Objetivos de Desarrollo Sostenible (ODS)

- **ODS 7: Energía asequible y no contaminante**
  – Al promover el uso eficiente de la iluminación mediante sensores y control inteligente, se contribuye a reducir el consumo energético y fomentar energías limpias.
- **ODS 11: Ciudades y comunidades sostenibles**
  – El proyecto aporta a crear ciudades más inteligentes, seguras y eficientes en el uso de recursos, mejorando la calidad de vida urbana en Tijuana.
- **ODS 9: Industria, innovación e infraestructura**
  – Impulsa la innovación tecnológica y la infraestructura digital para el desarrollo sostenible mediante IoT y soluciones conectadas en la nube.

Crecimiento urbano y demanda de servicios en Tijuana

- Tijuana es una de las ciudades de mayor crecimiento en México, con una población estimada de **1.9 millones de habitantes** en 2023 (INEGI, 2023). Este crecimiento genera una creciente demanda de infraestructura y servicios eficientes, incluyendo energía y tecnología inteligente. Fuente: INEGI - Censo de Población y Vivienda 2020

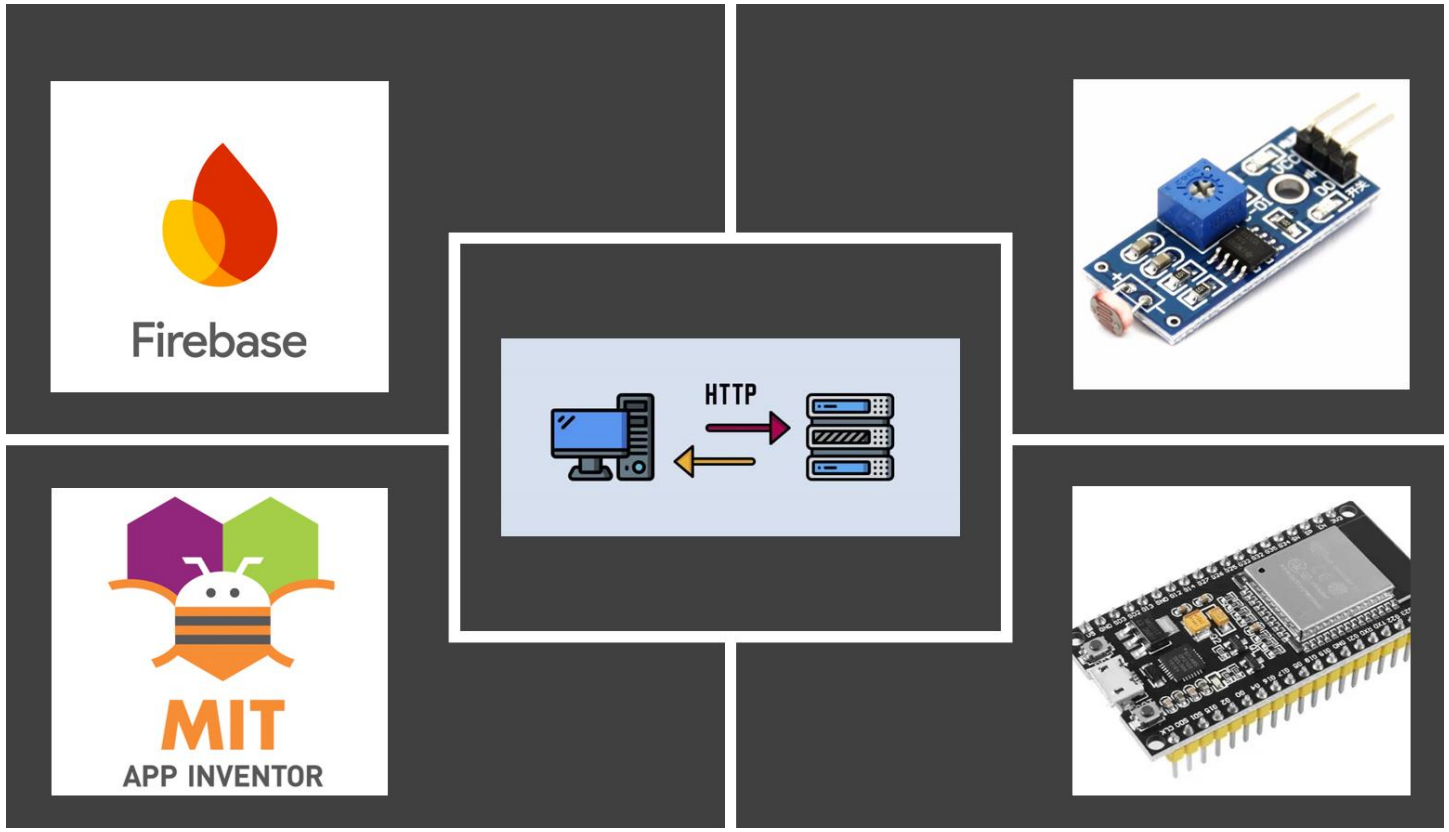
Uso de tecnologías IoT en México

- El mercado de IoT en México crece a un ritmo anual del **20%**, impulsado por la adopción en sectores residenciales y urbanos, según un reporte de Statista (2024). Esto muestra el potencial y relevancia de proyectos basados en IoT como el que se propone.
- Fuente: Statista - Internet of Things (IoT) in Mexico, 2024, Análisis de mercado IoT,

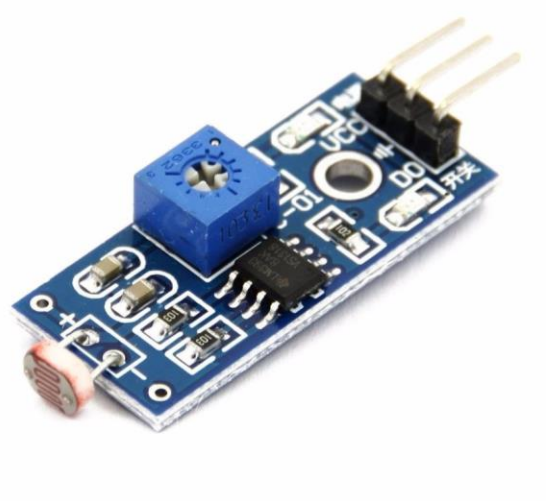
Problema a resolver

- En Tijuana, el consumo de energía en los hogares es alto y poco eficiente debido a la falta de sistemas inteligentes que permitan controlar el uso de dispositivos eléctricos.
- Esto genera desperdicio de energía y gastos innecesarios.
- No existen soluciones accesibles y prácticas que permitan monitorear el nivel de luz ambiental para encender o apagar dispositivos como un LED de manera automática y enviar esos datos para su análisis remoto.

Propuesta de solución



- Se desarrollará un sistema IoT con **ESP32** y una **fotoresistencia** para medir la luz ambiental y controlar un **LED** según el nivel detectado.
- El ESP32 enviará los datos a **Firebase** Realtime Database mediante **HTTP** para almacenarlos y consultarlos.
- Una aplicación móvil creada con **App Inventor** mostrará en tiempo real la intensidad de luz y el estado del LED, además de graficar los datos históricos.
- Así, se automatiza el control de la iluminación, se registran datos para análisis y se integra hardware, software y desarrollo móvil en un solo proyecto educativo.



Hardware

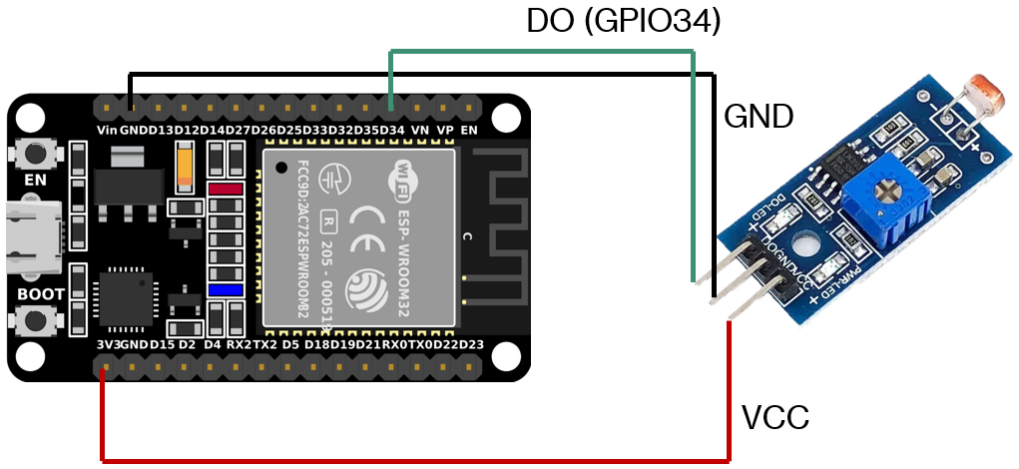
Material de Hardware	Cantidad	Descripción / Uso
ESP32	1	Microcontrolador con WiFi
Fotoresistencia	1	Sensor para medir luz ambiental
Resistencia 10 kΩ	1	Para conectar en serie con fotoresistencia
LED	1	Indicador visual (enciende/apaga)
Resistencia 220 Ω	1	Para proteger el LED
Protoboard	1	Para montar el circuito
Cables jumper	5-10	Para realizar las conexiones
Fuente de alimentación / USB	1	Para alimentar el ESP32

Software

Software	Cantidad	Descripción / Uso
Arduino IDE	1	Entorno de programación para ESP32
Librería WiFi para ESP32	1	Gestión de conexión WiFi en ESP32
Librería HTTPClient para ESP32	1	Para hacer peticiones HTTP/HTTPS desde el ESP32
Firebase Realtime Database	1	Base de datos en la nube para almacenamiento
Navegador Web	1	Para acceder a consola Firebase y monitoreo
Aplicación móvil (App Inventor)	1	Para visualizar datos y control desde móvil

Armado del circuito físico (fotoresistencia + LED + ESP32)

- **Paso 1:** Reúne los materiales
- **Paso 2:** Conecta la fotoresistencia (LDR)
  - Coloca la fotoresistencia en la protoboard.
  - Conecta un extremo de la fotoresistencia a 3.3 V del ESP32.
  - Conecta el otro extremo de la fotoresistencia al pin **GPIO34** (entrada analógica).
  - Esto crea un divisor de voltaje para que el ESP32 pueda leer el valor analógico.
- **Paso 3:** Probamos el código



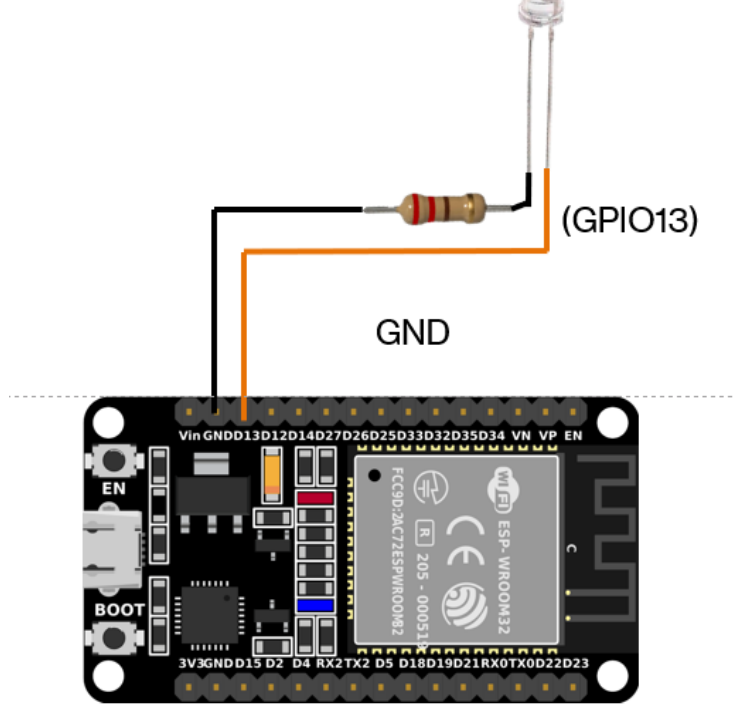
Salida Monitor Serie ×

Mensaje (Intro para mandar el mensaje de 'ESP32 Dev Modu

```
Valor de luz: 38
Valor de luz: 22
Valor de luz: 36
Valor de luz: 16
Valor de luz: 46
Valor de luz: 23
Valor de luz: 10
Valor de luz: 24
Valor de luz: 32
Valor de luz: 33
Valor de luz: 43
```

Lectura de luz y control de LED con ESP32

- **Paso 1:** Reúne los materiales
- **Paso 2:** Armado del circuito
  - Conecta el LED:
    - Ánodo (+) del LED → GPIO13 del ESP32 (u otro pin digital)
    - Cátodo (–) del LED → Resistencia de 220Ω → GND del ESP32
- **Paso 3:** probamos el código



Salida Monitor Serie ×

Mensaje (Intro para mandar el mensaje de 'ESP32

```
Oscuridad → LED apagado
Luz detectada (valor): 30
Oscuridad → LED apagado
Luz detectada (valor): 21
Oscuridad → LED apagado
Luz detectada (valor): 5
Oscuridad → LED apagado
Luz detectada (valor): 43
Oscuridad → LED apagado
Luz detectada (valor): 11
Oscuridad → LED apagado
```

Enviar datos del ESP32 a Firebase Realtime Database usando HTTP POST

Paso 1: Crear el proyecto en Firebase

- Ve a Firebase Console
- Haz clic en "**Agregar proyecto**" (o "Add project").
- Escribe el nombre del proyecto (por ejemplo: EcoLuzBC).
- Da clic en **Continuar**.
- Desactiva Google Analytics para este proyecto (opcional) y da clic en Crear proyecto.
- Espera a que se configure, luego da clic en **Continuar** para ir al panel.

Paso 2: Configurar Realtime Database

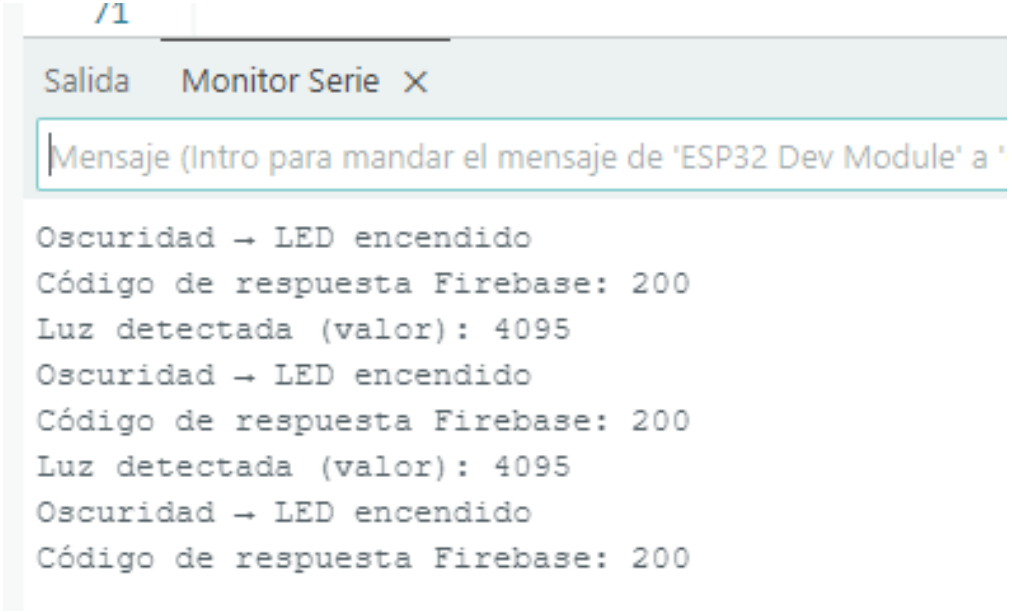
- En el menú lateral izquierdo, selecciona **Realtime Database**.
- Haz clic en **Crear base de datos**.
- Selecciona la ubicación más cercana a ti (por ejemplo: us-central1 o la que te ofrezca Firebase).
- Elige iniciar en modo **Prueba** (esto hará que la base esté abierta para leer y escribir sin autenticación, solo para desarrollo).
- Da clic en **Habilitar**.
- En la parte superior, verás la URL de tu base, algo como: https://monitoreoambientalbc-default-rtdb.firebaseio.com/
- **Guárdala, la necesitarás para tu ESP32.**

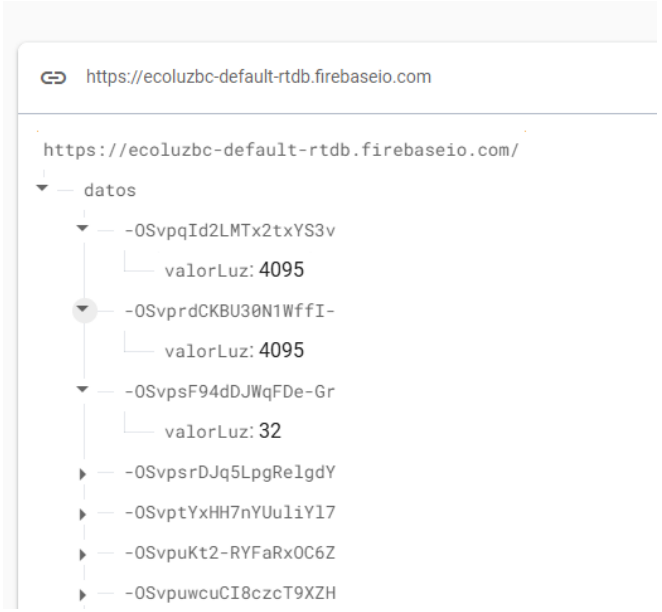
Paso 3: Configurar reglas de acceso

- Ve a la pestaña **Reglas** dentro de Realtime Database.
- Asegúrate que esté así mientras haces pruebas:

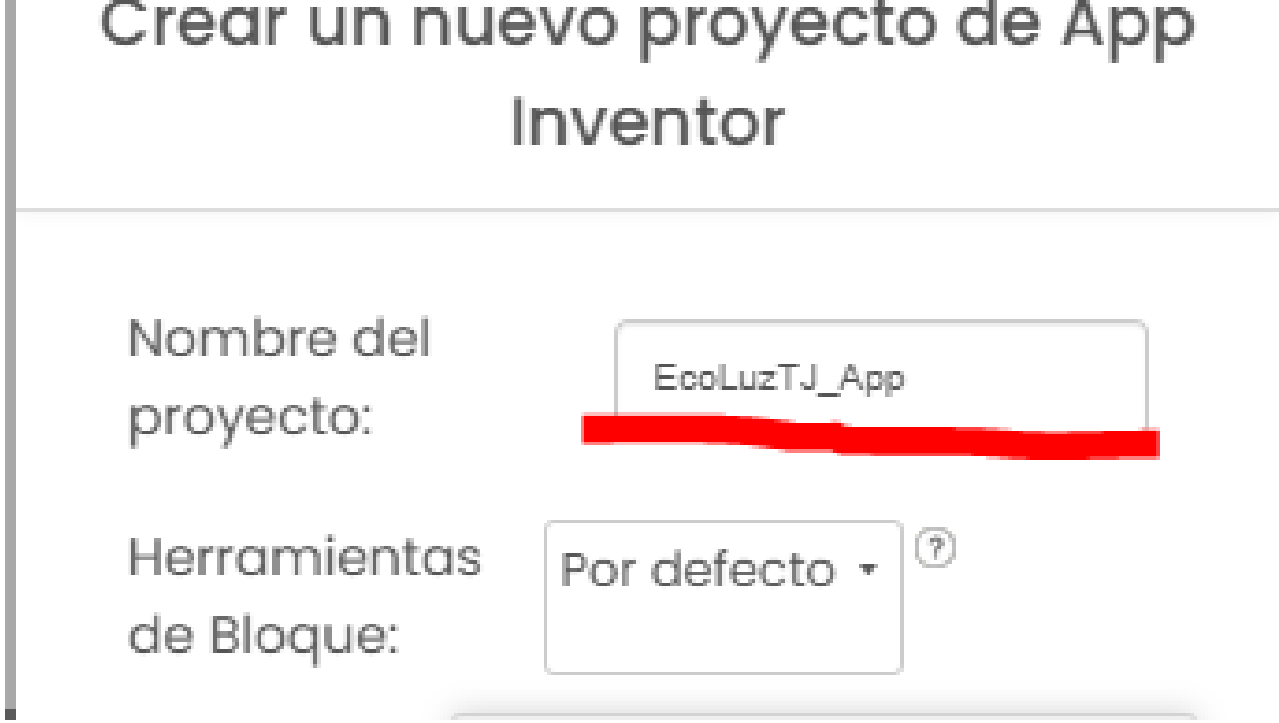
Paso 4: Probar código

- Carga el código al ESP32
  - Código
- Abre el Monitor Serial
- Verás mensajes
- **Verifica el encendido del LED**
  - Si el valor de luz es **mayor al umbral (1000)**, el LED se **enciende**.
  - Si es **menor**, el LED se **apaga**.
  - Puedes cubrir la fotoresistencia con tu mano o una hoja para probar la reacción del LED.
- **Revisa los datos en Firebase**
  - Entra a Firebase Console.
  - Abre tu proyecto EcoLuzBC.
  - Ve a la sección "Realtime Database" desde el menú lateral.
  - Asegúrate de que el enlace termine en .json internamente.
  - Verás los datos guardados así:



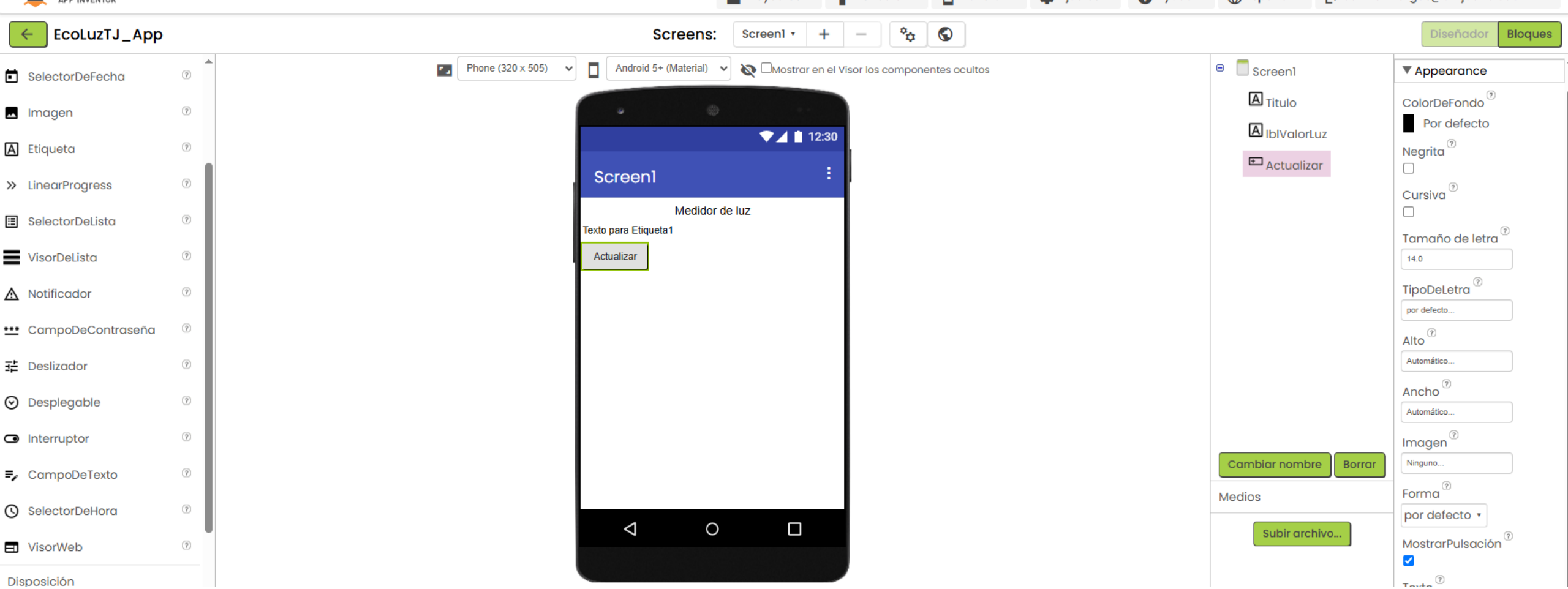


Crear la app móvil en App Inventor para mostrar los datos de luz desde Firebase



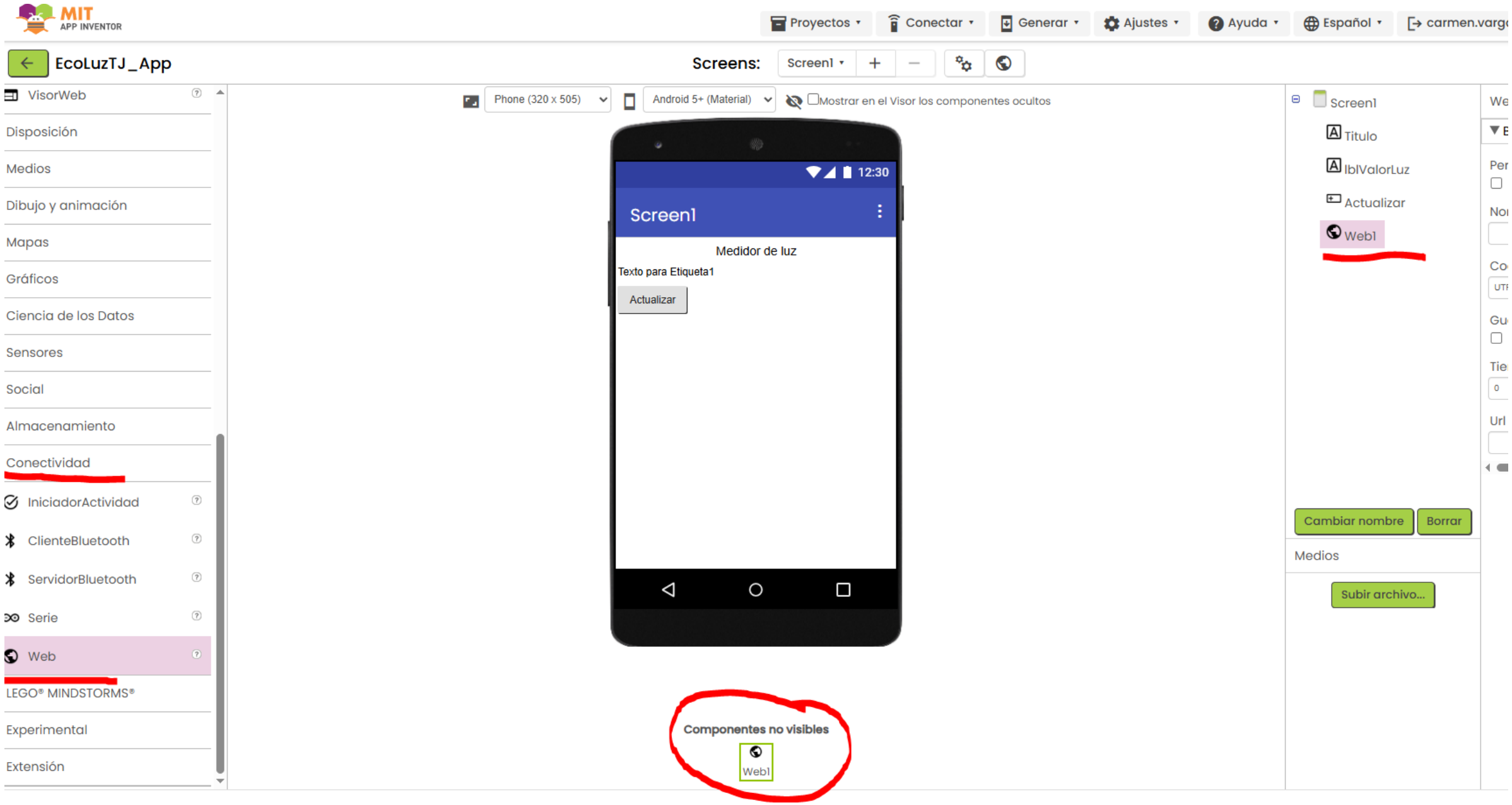
1. Entra a App Inventor

1. Ve a: https://ai2.appinventor.mit.edu
2. Inicia sesión con tu cuenta de Google.
3. Haz clic en **"Create new project"** y nómbralo: **EcoLuzTJ_App**



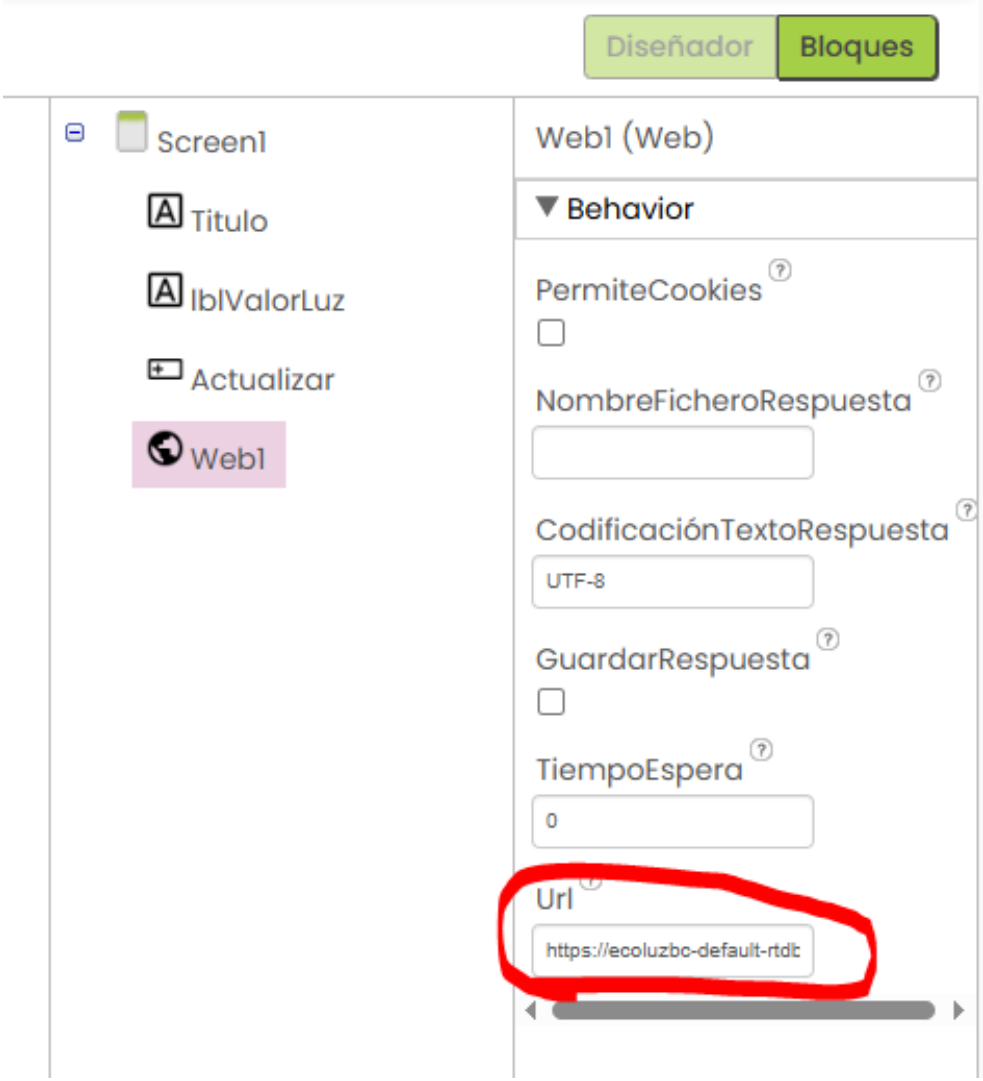
2. Agrega los siguientes componentes en la pantalla

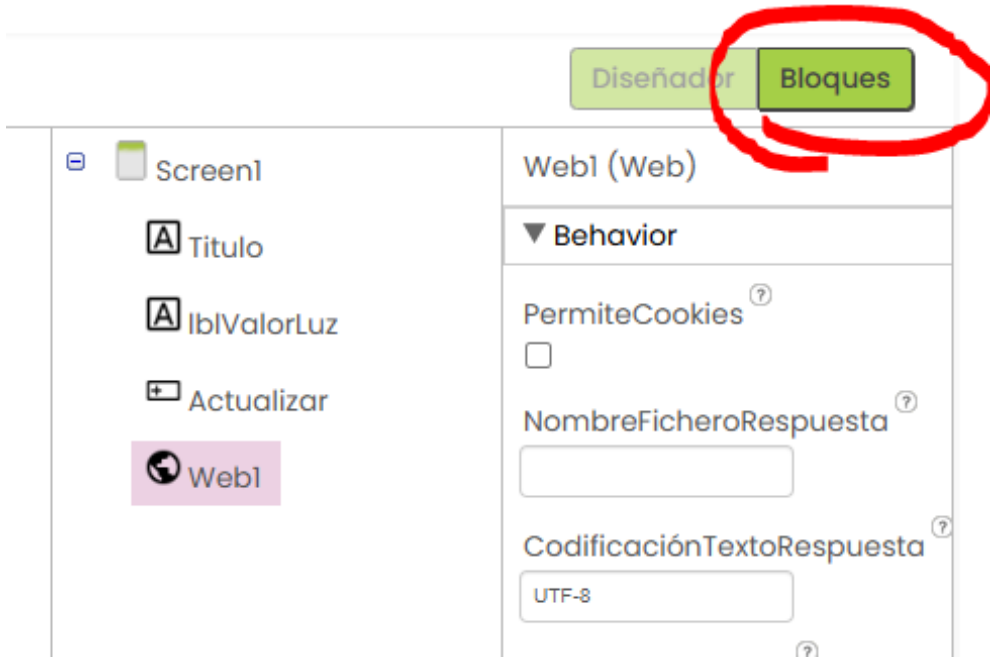
- Desde "User Interface":
  - **1 Label** → para mostrar el título ("Medidor de luz")
  - **1 Label** → para mostrar el valor leído de Firebase
  - **1 Button** → para actualizar el dato manualmente (texto: “Actualizar”)
- Desde "Connectivity":
  - 1 componente **Web** (lo llamaremos Web1)


MIT APP INVENTOR
Proyectos
Conectar
Generar
Ajustes
Ayuda
Español
carmen.varg
EcoLuzTJ_App
Screens:
Screen1
VisorWeb
Disposición
Medios
Dibujo y animación
Mapas
Gráficos
Ciencia de los Datos
Sensores
Social
Almacenamiento
Conectividad
IniciadorActividad
ClienteBluetooth
ServidorBluetooth
Serie
Web
LEGO® MINDSTORMS®
Experimental
Extensión
Phone (320 x 505)
Android 5+ (Material)
Mostrar en el Visor los componentes ocultos
12:30
Screen1
Medidor de luz
Texto para Etiqueta1
Actualizar
Componentes no visibles
Web1
Screen1
Titulo
lblValorLuz
Actualizar
Web1
Cambiar nombre
Borrar
Medios
Subir archivo...

3. Configura el componente Web

- selecciona **Web1** y en las propiedades pon esto:
  - https://ecoluzbc-default-rtdb.firebaseio.com/luz.json
- Asegúrate de que la ruta luz es **la misma** donde el ESP32 está guardando el dato en **Firebase**.

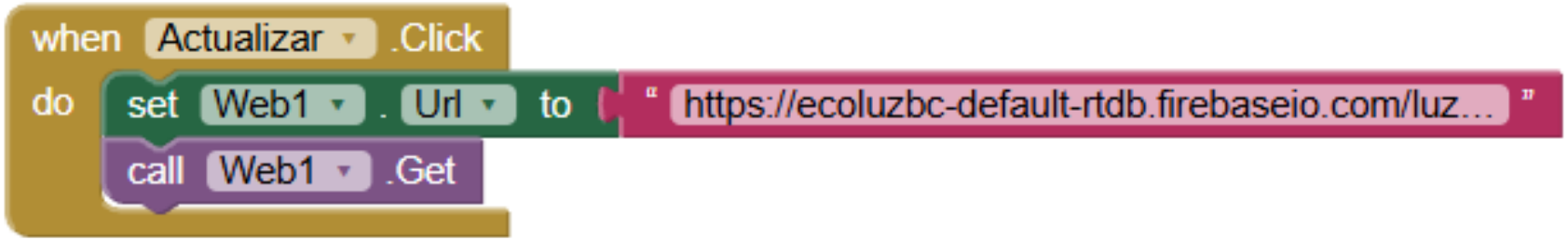




4. Programa los bloques

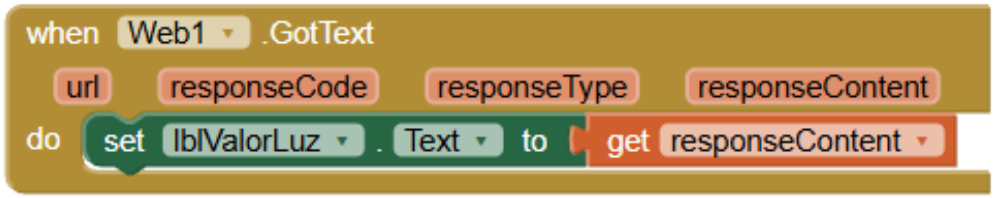
1. **Haz clic en la pestaña "Blocks" (Bloques)** en la parte superior derecha de App Inventor.

4. Programa los bloques



1. **Cuando se presione el botón**
   - Busca el bloque:
     - when Button1.Click do
   - Dentro de ese bloque, agrega:
     - set Web1.Url to https://ecoluzbc-default-rtdb.firebaseio.com/luz.json
     - call Web1.Get

4. Programa los bloques



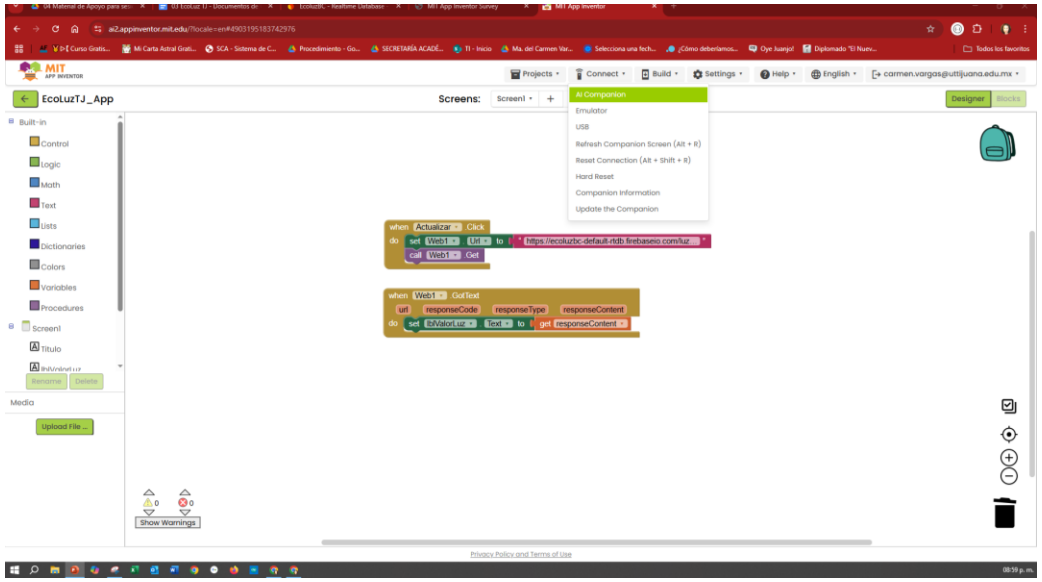
2. **Cuando reciba la respuesta:**
   - Busca el bloque:
     - when Web1.GotText do
   - Dentro de ese bloque, agrega:
     - set **Label2.Text** to get responseContent

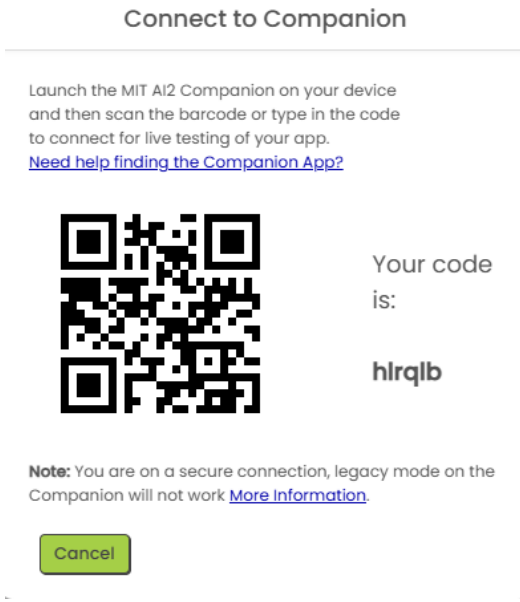
5. Instala la app de prueba en tu celular

- En tu celular Android:
  - Descarga MIT AI2 Companion desde Google Play Store.
- Abre la app y déjala lista.

6. Prueba tu app en tiempo real (Live Test)

- En tu compu, en App Inventor:
  - Haz clic en el menú **"Connect"** → **"AI Companion“**
  - Verás un código **QR**
  - En tu celular, abre la app AI2 Companion y **escanea ese código**
- Espera unos segundos... tu app aparecerá en tu celular y funcionará en tiempo real.





7. Pulsa el botón "Actualizar" en la app

- La app debería conectarse a Firebase
- Leer el valor que se está guardando (por el ESP32)
- Y mostrarlo en Label2.

¿Qué necesitas tener listo antes de probar?

- ESP32 encendido y **enviando datos a Firebase**.
- En Firebase, en tu base de datos en tiempo real, debe existir un campo llamado luz y tener algún valor.
- El ESP32 debe estar enviando datos con la ruta correcta

Adaptar

Usa **POST** si quieres guardar varios valores diferentes (como un registro histórico).

Usa **PUT** si solo quieres guardar el último valor y reemplazar el anterior.

Adaptación

- Ejemplo completo para que el ESP32 envíe el valor del sensor usando HTTP PUT, así siempre se sobrescribe y solo tendrás un valor guardado
- Código