Documento Maestro: App "Ubicador de Bultos"

Propsito: Dejar un brief completo y detallado para un Project Manager y el equipo de desarrollo sobre la webapp de ubicacin de bultos en depsito.

---

1. Visin General

Una webapp que permita almacenar y actualizar en tiempo real la posicin de bultos dentro de un depsito, usando lectores de cdigos (teclado o cmara). Los datos se guardan en una base de datos MongoDB con retencin de 45 das y se accede mediante una API REST.

Dispositivos objetivo:
- iPhone (navegador mvil en modo kiosk)
- Terminal Skorpio de Hasar (navegador equipado en PDA)

Ruta de despliegue en VM: /var/www/html/via/ubicador
SO VM: Ubuntu (20.04/22.04)

---

2. Flujo de Usuario

1. Usuario abre la webapp en modo kiosk (sin scroll).
2. Aparece mensaje: "Por favor escanear una ubicacin para comenzar".
3. Escanea con lector (teclado+enter) o toca cono de cmara para usar la cmara.
4. Si el texto escaneado empieza con UBI , la app extrae la ubicacin (p.ej. A1), la muestra en la parte superior y cambia mensaje a "Escanea un bulto para ubicarlo".
5. Al escanear un cdigo de bulto (p. ej. 040000094059990283188690 01):
   - Descartar los primeros 7 dgitos y los siguientes 4.
   - Extraer:
     - Gua: dgitos 9405? 999028318869

- Bulto: ltimos 3 dgitos 001

6. Enviar POST a la API:

POST /api/ubicaciones

{

"guia": "999028318869",

"bulto": "001",

"ubicacion": "A1",

"fecha": "DD/MM/YYYY HH:mm"

}

7. Si la API confirma xito:

- Sonido de xito y flash verde de pantalla.

- Mensaje: "Bulto 001 de la gua 999028318869: Ubicado en A1".

8. Si la API falla:

- Reintentar hasta 3 veces.

- Al tercer fallo, sonido de error, flash rojo y mensaje: "Error. No se pudo guardar la ubicacin del bulto."

9. Al desconexin de Internet:

- Mensaje persistente "Sin conexin" y botn "Reintentar".

10. Escanear fuera de formato (ni UBI ni bulto vlido):

- Flash rojo y mensaje: "Error: cdigo no reconocido" + sonido de error.

---

3. Requisitos Funcionales

- Escaneo mixto: teclado acta como input + botn de cmara.
- Validacin de formato:
  - Ubicacin: escaneo comienza con UBI .
  - Bulto: 26 caracteres, validacin de posicin y longitud mnima.
- Feedback inmediato: colores (verde/rojo), sonidos (xito/fallo) y mensajes.
- 3 reintentos ante fallo de guardado.
- Gestin de conexin: mensaje y retry.
- Borrado automtico: retencin 45 das.

- Sobrescritura de ubicacin si reaparece el mismo bulto.
- Sin login, autenticacin va x-api-key.

---

4. Requisitos No Funcionales

- Disponibilidad permanente (200 OK en escaneo real).
- Escalabilidad primaria para ~4.000 bultos/mes.
- Sin scroll en pantalla (kiosk).
- Tipografa: Cairo.
- Frameworks: Bootstrap + FontAwesome.

---

5. Arquitectura y Stack

- Frontend:
  - Single HTML view con Bootstrap, FontAwesome, fuente Cairo.
  - JavaScript puro (ES6+).
- Backend: Node.js + Express.
- DB: MongoDB Community Edition (misma VM).
- Proxy y procesos: Nginx + PM2.
- Entorno: variables en .env.

---

6. Esquema de la Base de Datos

Coleccin: guias

Cada documento agrupa todos los bultos de una misma gua en un array de subdocumentos.

Ejemplo de documento:

```
{
  "guia": "999012345678",
  "bultos": [
    {
      "bulto": "001",
      "ubicacion": "A1",
      "fecha": "27/07/2025 16:31",
      "fecha_iso": "2025-07-27T13:31:00Z"
    },
    {
      "bulto": "002",
      "ubicacion": "E3",
      "fecha": "24/07/2025 12:35",
      "fecha_iso": "2025-07-24T09:35:00Z"
    }
  ]
}
```

Retencin de datos (45 das)

MongoDB no permite aplicar un ndice TTL a campos dentro de subdocumentos sin eliminar todo el documento. Para garantizar que cada bulto expire a los 45 das:

1. Se crea un cron job en el backend que se ejecuta diariamente.
2. Ese job recorre la coleccin guias y elimina de su array bultos aquellos subdocumentos cuyo campo fecha_iso sea >45 das.
3. Si, tras la limpieza, un documento guia queda con el array bultos vaco, se elimina por completo.

---

7. API REST

Endpoints

1. POST /api/ubicaciones
   - Headers: Content-Type: application/json, x-api-key: <WRITE_KEY>
   - Body: { "guia": "string", "bulto": "string", "ubicacion": "string", "fecha": "DD/MM/YYYY HH:mm" }
   - Respuestas:
     - 201 Created -> { "status": "ok" }
     - 400 Bad Request -> { "error": "mensaje" }
     - 401 Unauthorized -> { "error": "API key invalida" }
     - 500 Internal Error

2. GET /api/ubicaciones/:guia (opcional)
   - Headers: x-api-key: <READ_KEY>
   - Respuesta: { "guia": "999...", "bultos": [{ "bulto":"001", "ubicacion":"A1", "fecha":"..." }, ...] }

---

8. Seguridad

- API Keys en variables de entorno: API_KEY_WRITE, API_KEY_READ
- Middleware Express verifica req.headers['x-api-key'].

---

9. Deployment en VM Ubuntu

1. Instalar dependencias:
   apt update && apt install -y nodejs npm mongodb nginx git
2. Clonar repositorio en /var/www/html/via/ubicador.
3. Variables de entorno en /var/www/html/via/ubicador/.env:
   API_KEY_WRITE=...
   API_KEY_READ=...
   MONGO_URI=mongodb://localhost:27017/ubicador
   TZ=America/Argentina/Buenos_Aires
4. Instalar y arrancar Node/Express con PM2:

cd /var/www/html/via/ubicador

npm install

pm2 start src/app.js --name ubicador-api

pm2 save

5. Configurar Nginx (/etc/nginx/sites-available/ubicador.conf):

server { listen 80; server_name _; root /var/www/html/via/ubicador/public; index index.html;

location /api/ { proxy_pass http://localhost:3000/; }

location / { try_files $uri $uri/ =404; }

}

ln -s ... && nginx -t && systemctl reload nginx

---

10. Archivos y Recursos

/var/www/html/via/ubicador

public/

index.html

css/

js/

sounds/

src/

app.js

ecosystem.config.js

.env

package.json

---

11. Prximos Pasos

- Elegir puerto de Express (3000 por defecto) y configurar PM2.
- Crear certificado TLS en futuro.

- Desarrollar pruebas unitarias y de extremo a extremo.
- Plan de monitoreo (logs, alertas).