Endpoits

Modelos de desarrollo web

Liseth Danae Mecatl Toxcoyoa

1. Listar todos los deportes individuales

@app.get("/deportes/tipo/individual")

async def deportes\_individuales():

return get\_hierarchy\_level("Deportes", "PorTipo", "Individual")

2. Listar todos los deportes de equipo

@app.get("/deportes/tipo/equipo")

async def deportes\_equipo():

return get\_hierarchy\_level("Deportes", "PorTipo", "Equipo")

3. Deportes de alta intensidad

@app.get("/deportes/intensidad/alta")

async def deportes\_alta\_intensidad():

return get\_hierarchy\_level("Deportes", "PorIntensidad", "Alta")

4. Deportes de media intensidad

@app.get("/deportes/intensidad/media")

async def deportes\_media\_intensidad():

return get\_hierarchy\_level("Deportes", "PorIntensidad", "Media")

5. Deportes de baja intensidad

@app.get("/deportes/intensidad/baja")

async def deportes\_baja\_intensidad():

return get\_hierarchy\_level("Deportes", "PorIntensidad", "Baja")

6. Deportes de interior

@app.get("/deportes/ambiente/interior")

async def deportes\_interior():

return get\_hierarchy\_level("Deportes", "PorAmbiente", "Interior")

7. Deportes de exterior

@app.get("/deportes/ambiente/exterior")

async def deportes\_exterior():

return get\_hierarchy\_level("Deportes", "PorAmbiente", "Exterior")

8. Deportes mixtos (interior/exterior)

@app.get("/deportes/ambiente/mixto")

async def deportes\_mixto():

return get\_hierarchy\_level("Deportes", "PorAmbiente", "Mixto")

9. Obtener alumnos por deporte específico

@app.get("/alumnos/deporte/{deporte}")

async def alumnos\_por\_deporte(deporte: str):

resultados = [a for a in alumnos if a["Deporte favorito"].lower()

== deporte.lower()]

if not resultados:

raise HTTPException(

status\_code=404, detail=f"No se encontraron alumnos con deporte favorito: {deporte}")

return resultados

10. Estadísticas generales

@app.get("/deportes/estadisticas")

async def estadisticas():

conteo = {}

for alumno in alumnos:

deporte = alumno["Deporte favorito"]

if deporte:

conteo[deporte] = conteo.get(deporte, 0) + 1

return {

"total\_alumnos": len(alumnos),

"deportes\_unicos": len(conteo),

"alumnos\_por\_deporte": conteo,

"deporte\_mas\_popular": max(conteo, key=conteo.get) if conteo else None

}