

**CSC-640-MI**

## **Telemetry Ingestion & Analysis API**

A lightweight API for iRacing-style IBT files

# Project Overview

## Purpose:

- Upload telemetry files from iRacing simulator
- Store and query session metadata, weather, and driver data
- Extract lap-by-lap telemetry attributes
- Compute per-lap metrics and statistics

## Tech Stack:

- PHP with FastRoute
- MySQL for relational data
- Python for IBT file parsing
- Docker containerized

# Architecture

## Components:

- `TelemetryController` - handles file uploads
- `SessionController` - CRUD for sessions and laps
- `LapService` - lap boundary detection and incident tracking
- Python parser - extracts attributes from `.ibt` files

## Database Tables:

- `session_info` - session metadata
- `weather` - track conditions
- `driver` - driver & car info
- `attribute_values` - time-series telemetry data

# API Endpoints Overview

1. **POST** /api/telemetry/upload - Upload IBT file
2. **GET** /api/sessions - List all sessions
3. **GET** /api/sessions/{id} - Get session details
4. **GET** /api/sessions/{id}/laps - Get lap list with incidents
5. **GET** /api/sessions/{id}/laps/{lapNumber} - Get lap attribute data
6. **GET** /api/sessions/{id}/laps/{lapNumber}/averages - Get lap averages
7. **DELETE** /api/sessions/{id}/laps/{lapNumber} - Delete lap data
8. **DELETE** /api/sessions/{id} - Delete full session

# 1. Upload Telemetry

**Endpoint:** POST /api/telemetry/upload

**Purpose:** Upload an .ibt file and extract specified attributes

**Request:**

```
curl --request POST \
--url http://localhost/api/telemetry/upload \
--header 'content-type: multipart/form-data' \
--form 'telemetry_file=@telemetry/porsche992rgt3_roadatlanta_full_test1.ibt' \
--form 'attributes[]="Lap"' \
--form 'attributes[]="RPM"' \
--form 'attributes[]="Speed"' \
--form 'attributes[]="LapDistPct"' \
--form 'attributes[]="FuelLevel"' \
--form 'attributes[]="RFpressure"' \
--form 'attributes[]="RRpressure"' \
--form 'attributes[]="LFpressure"' \
--form 'attributes[]="LRpressure"' \
--form 'attributes[]="PlayerIncidents'"
```

# Upload Response

Example Response:

```
{"uploaded":true,"session_id":"2478c41b-dceb-449e-9b97-a911050d276b"}
```

## 2. List All Sessions

**Endpoint:** GET /api/sessions

**Purpose:** Retrieve all stored telemetry sessions

**Request:**

```
curl http://localhost/api/sessions
```

# List Sessions Response

## Example Response:

```
{  
  "sessions": [  
    {  
      "session_id": "2478c41b-dceb-449e-9b97-a911050d276b",  
      "session_type": "Offline Testing",  
      "track_name": "Road Atlanta",  
      "track_id": 127,  
      "track_config": "Full Course",  
      "session_date": "2025-10-25",  
      "session_time": "9:35 am",  
      "track_config_sector_info": "[{\\"SectorNum\\": 0, \\"SectorStartPct\\": 0.0}, {\\"SectorNum\\": 1, \\"SectorStartPct\\": 0.167875}, {\\"SectorNum\\": 2, \\"SectorStartPct\\": 0.442307}, {\\"SectorNum\\": 3, \\"SectorStartPct\\": 0.787105}]"  
    },  
    {  
      "session_id": "77c222ac-4514-408d-aafb-7b92f8ccab3e",  
      "session_type": "Offline Testing",  
      "track_name": "Road Atlanta",  
      "track_id": 127,  
      "track_config": "Full Course",  
      "session_date": "2025-10-25",  
      "session_time": "9:35 am",  
      "track_config_sector_info": "[{\\"SectorNum\\": 0, \\"SectorStartPct\\": 0.0}, {\\"SectorNum\\": 1, \\"SectorStartPct\\": 0.167875}, {\\"SectorNum\\": 2, \\"SectorStartPct\\": 0.442307}, {\\"SectorNum\\": 3, \\"SectorStartPct\\": 0.787105}]"  
    }  
  ]  
}
```

### 3. Get Single Session

**Endpoint:** GET /api/sessions/{id}

**Purpose:** Get detailed session info including weather and drivers

**Request:**

```
curl http://localhost/api/sessions/2478c41b-dceb-449e-9b97-a911050d276b
```

# Session Details Response

## Example Response:

```
{  
  "session_info": {  
    "session_id": "2478c41b-dceb-449e-9b97-a911050d276b",  
    "session_type": "Offline Testing",  
    "track_name": "Road Atlanta",  
    "track_id": 127,  
    "track_config": "Full Course",  
    "session_date": "2025-10-25",  
    "session_time": "9:35 am",  
    "track_config_sector_info": "[{\\"SectorNum\\": 0, \\"SectorStartPct\\": 0.0}, {\\"SectorNum\\": 1, \\"SectorStartPct\\": 0.167875}, {\\"SectorNum\\": 2, \\"SectorStartPct\\": 0.442307}, {\\"SectorNum\\": 3, \\"SectorStartPct\\": 0.787105}]"  
  },  
  "weather": {  
    "session_id": "2478c41b-dceb-449e-9b97-a911050d276b",  
    "track_air_temp": "18.90 C",  
    "track_surface_temp": "18.90 C",  
    "track_precipitation": "0 %",  
    "track_fog_level": "0 %",  
    "track_wind_speed": "3.22 km/h",  
    "track_wind_direction": "N"  
  },  
  "drivers": [  
    {  
      "session_id": "2478c41b-dceb-449e-9b97-a911050d276b",  
      "driver_user_id": 1159240,  
      "driver_name": "Mitchell Isler",  
      "car_number": "64",  
      "car_name": "Porsche 911 GT3 R (992)",  
      "car_class_id": 0,  
      "driver_rating": 1  
    }  
  ]  
}
```

## 4. Get Lap List & Counts

**Endpoint:** GET /api/sessions/{id}/laps

**Purpose:** Get lap boundaries and incident detection per lap

**Request:**

```
curl http://localhost/api/sessions/{SESSION_ID}/laps
```

**Features:**

- Detects lap start/end indices
- Checks for incidents in each lap
- Returns valid\_lap boolean

# Lap List Response

## Example Response:

```
{  
  "session_id": "2478c41b-dceb-449e-9b97-a911050d276b",  
  "lap_count": 9,  
  "valid_lap_count": 9,  
  "invalid_lap_count": 0,  
  "laps": [  
    {  
      "lap_number": 1,  
      "start_index": 5441,  
      "end_index": 10421,  
      "sample_count": 4981,  
      "valid_lap": true,  
      "incidents_in_lap": null  
    },  
    {  
      "lap_number": 2,  
      "start_index": 10422,  
      "end_index": 15365,  
      "sample_count": 4944,  
      "valid_lap": true,  
      "incidents_in_lap": null  
    }  
  ]}
```

## 5. Get Lap Attribute Data

**Endpoint:** GET /api/sessions/{id}/laps/{lapNumber}

**Purpose:** Extract raw telemetry data for specific attributes over a lap

**Request:**

```
curl "http://localhost/api/sessions/{SESSION_ID}/laps/2?attribute=RPM"
```

**Returns:** Frame-by-frame data for requested attribute

# Lap Attribute Data Response

## Example Response:

```
{  
  "session_id": "e2874d50-9a11-4159-9ce8-f5add3669ac3",  
  "lap_number": "5",  
  "attribute": "RPM",  
  "start_index": 25145,  
  "end_index": 29973,  
  "sample_count": 4829,  
  "data": {  
    "25145": 8104.89013671875,  
    "25146": 8093.4599609375,  
    "25147": 8081.755859375,  
    "25148": 8095.7490234375,  
    "25149": 8116.71728515625,  
    "25150": 8116.158203125,  
    ...  
  }  
}
```

## 6. Get Lap Averages

**Endpoint:** GET /api/sessions/{id}/laps/{lapNumber}/averages

**Purpose:** Compute average, min, max for attributes over a lap

**Request:**

```
curl "http://localhost/api/sessions/{SESSION_ID}/laps/2/averages?attribute=RFPressure,LFPressure,RRPressure,LRPressure"
```

**Use Cases:**

- Compare lap performance
- Identify anomalies
- Summarize telemetry quickly

# Lap Averages Response

## Example Response:

```
{  
    "session_id": "e2874d50-9a11-4159-9ce8-f5add3669ac3",  
    "lap_number": "5",  
    "start_index": 25145,  
    "end_index": 29973,  
    "lap_sample_count": 4829,  
    "attributes": {  
        "RFPressure": {  
            "average": 171.6104196567026,  
            "min": 170.7393798828125,  
            "max": 172.33935546875,  
            "sample_count": 4829  
        },  
        "LFPressure": {  
            "average": 178.0650377429576,  
            "min": 176.93081665039062,  
            "max": 179.04281616210938,  
            "sample_count": 4829  
        },  
        "RRPressure": {  
            "average": 169.51825133493708,  
            "min": 168.7683563232422,  
            "max": 170.08563232421875,  
            "sample_count": 4829  
        }  
    }  
}
```

## 7. Delete Lap Attribute Data

**Endpoint:** `DELETE /api/sessions/{id}/laps/{lapNumber}`

**Purpose:** Remove telemetry data for specific lap

**Delete specific attributes:**

```
curl -X DELETE "http://localhost/api/sessions/{SESSION_ID}/laps/2?attribute=Speed"
```

**Delete ALL attributes for lap:**

```
curl -X DELETE "http://localhost/api/sessions/{SESSION_ID}/laps/2"
```

# Delete Lap Data Response

## Example Response:

```
{  
    "session_id": "e2874d50-9a11-4159-9ce8-f5add3669ac3",  
    "lap_number": "5",  
    "attributes_deleted": [  
        "FuelLevel",  
        "Lap",  
        "LapDistPct",  
        "LFpressure",  
        "LRpressure",  
        "OnPitRoad",  
        "PlayerIncidents",  
        "RFpressure",  
        "RPM",  
        "RRpressure",  
        "Speed"  
    ],  
    "start_index": 25145,  
    "end_index": 29973,  
    "data_points_deleted": 53119,  
    "message": "Successfully deleted attribute data for lap 5"  
}
```

## 8. Delete Full Session

**Endpoint:** `DELETE /api/sessions/{id}`

**Purpose:** Delete entire session and all associated data

**Request:**

```
curl -X DELETE "http://localhost/api/sessions/{SESSION_ID}"
```

**Cascades to:**

- Session info
- Weather data
- Driver records
- All attribute values

# Delete Session Response

Example Response:

```
{  
  "session_id": "e2874d50-9a11-4159-9ce8-f5add3669ac3",  
  "message": "Session and all associated data deleted successfully",  
  "deleted_records": {  
    "session_info": 1,  
    "weather": 1,  
    "drivers": 1,  
    "attribute_values": 11  
  }  
}
```

## Key Features

-  **Lap Detection** - Automatically identifies lap boundaries from telemetry
-  **Incident Tracking** - Flags laps with incidents ( `valid_lap` boolean)
-  **Flexible Queries** - Multi-attribute support via comma-separated or repeated params
-  **Statistical Analysis** - Built-in avg/min/max calculations
-  **Selective Deletion** - Delete individual laps or full sessions

# LapService Architecture

## Core Functionality:

- `getLapIndices()` - Parses "Lap" attribute to find lap boundaries
- `addIncidentData()` - Checks "PlayerIncidents" attribute frame-by-frame
- Returns structured lap data with start/end indices

## Reusable across endpoints:

- Lap list
- Attribute extraction
- Averages computation
- Deletion operations

# Demo & Testing

## Quick Start:

```
docker-compose up --build -d  
docker-compose run --rm migrate
```

## Test Resources:

- Example `.ibt` files in `telemetry/` directory
- Postman collection: `PostmanCollection.json`
- curl examples: `API_CURL.md`

# Future Enhancements

## Potential additions:

- OAuth/JWT authentication
- Real-time telemetry streaming
- Advanced analytics (sector times, tire degradation)
- Comparison tools (multiple laps/sessions)
- Export to CSV/visualization formats
- Web dashboard for data exploration

# Questions?

## Resources:

- GitHub: [CSC-640-MI](#)
- API Documentation: [API\\_CURL.md](#)
- Database Schema: [db/create\\_db.php](#)

Thank you!