

# Application for 2-Wheeler Rider Risk Estimation



# Problem Statement

The "2-Wheeler Rider Risk Estimation" app calculates a real-time risk score based on rider data like lean angle and cumulative speed. It helps identify high-risk scenarios to enhance safety and reduce accidents.



# Importance of the Problem

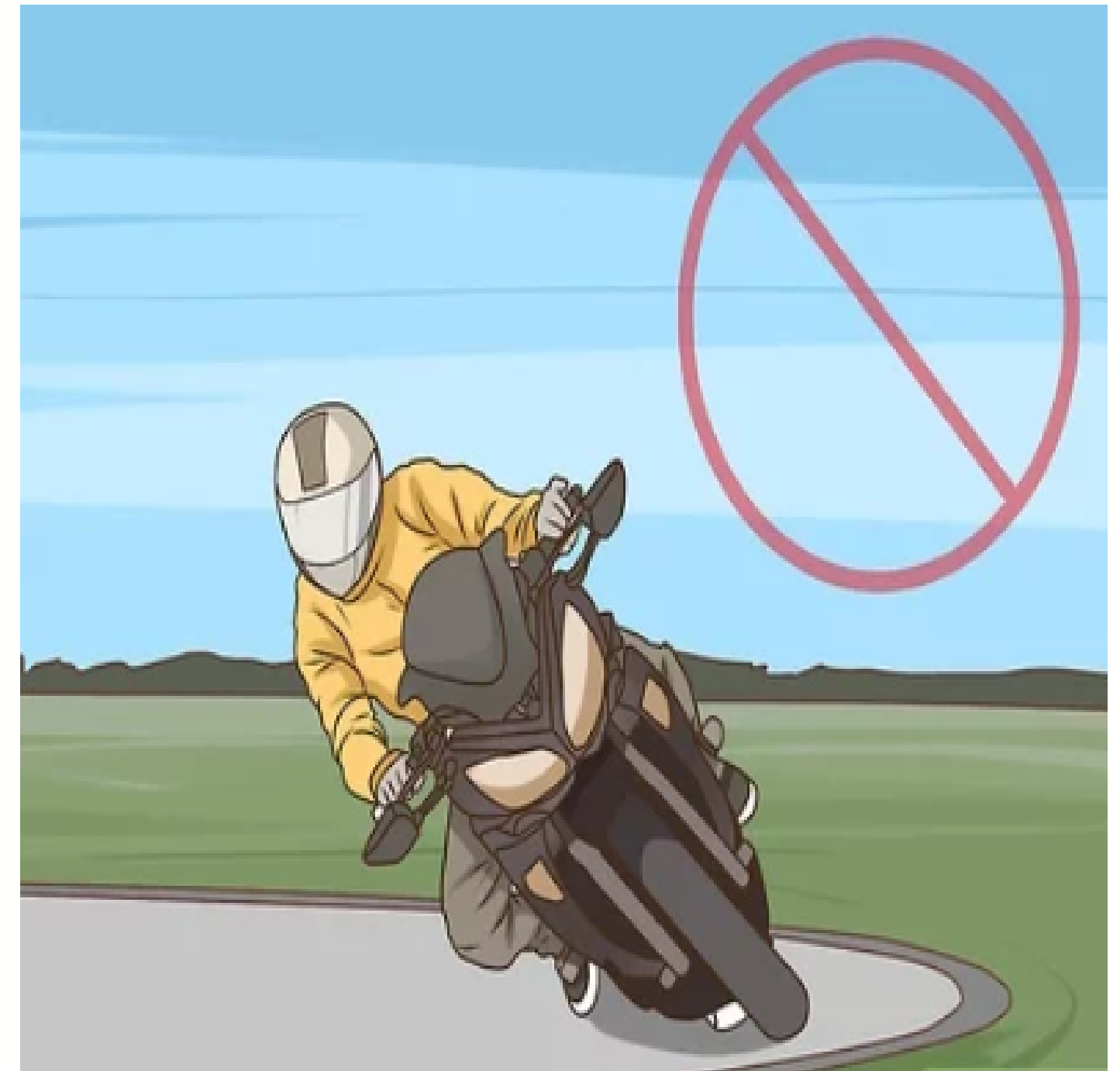
Enhance rider safety by reducing accidents in high-risk scenarios such as

- Skidding
- Collisions, and
- Loss of control.

Identify, quantify, and mitigate factors contributing to risky driving patterns using real-time feedback.

Provide insights based on lean angle and cumulative speed to help riders make safer decisions.

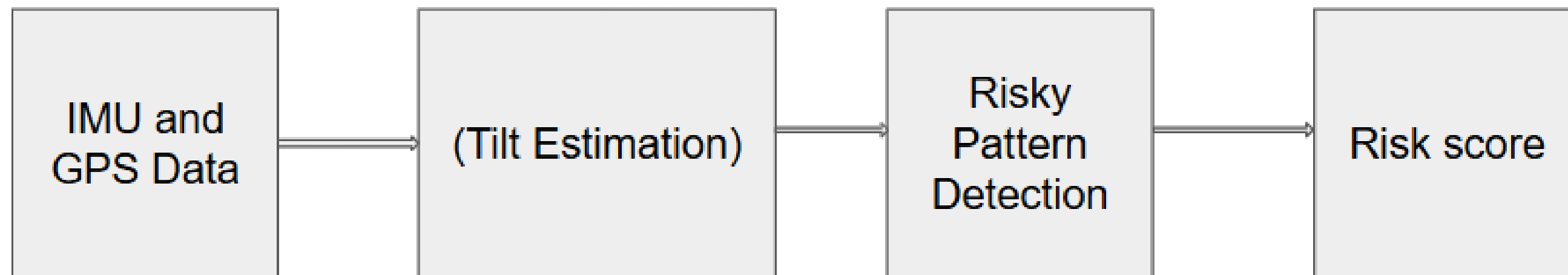
Address the growing need for improved safety measures in two-wheeler transportation.



# Objectives

- Tilt estimation
- Risky driving pattern detection
- Application Design

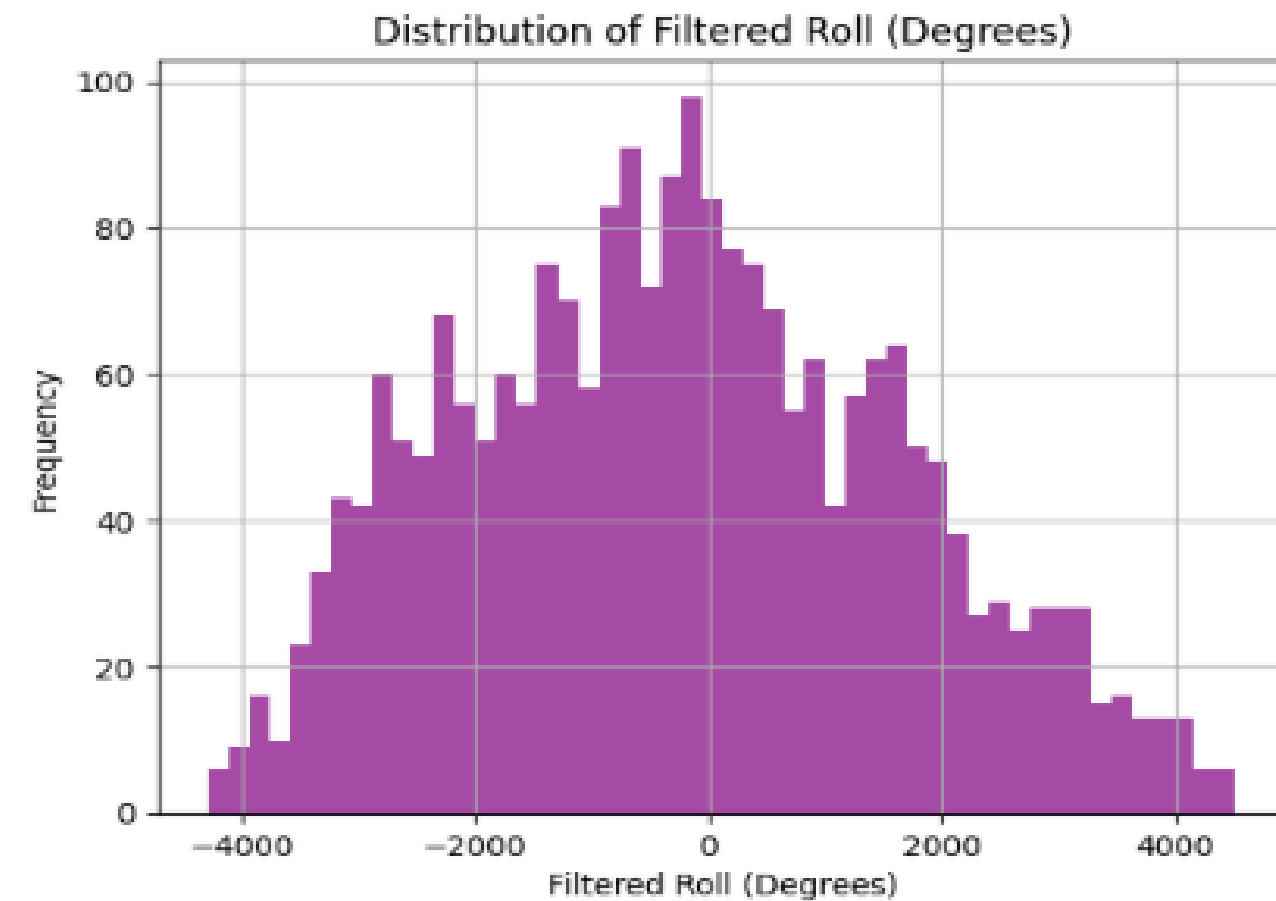
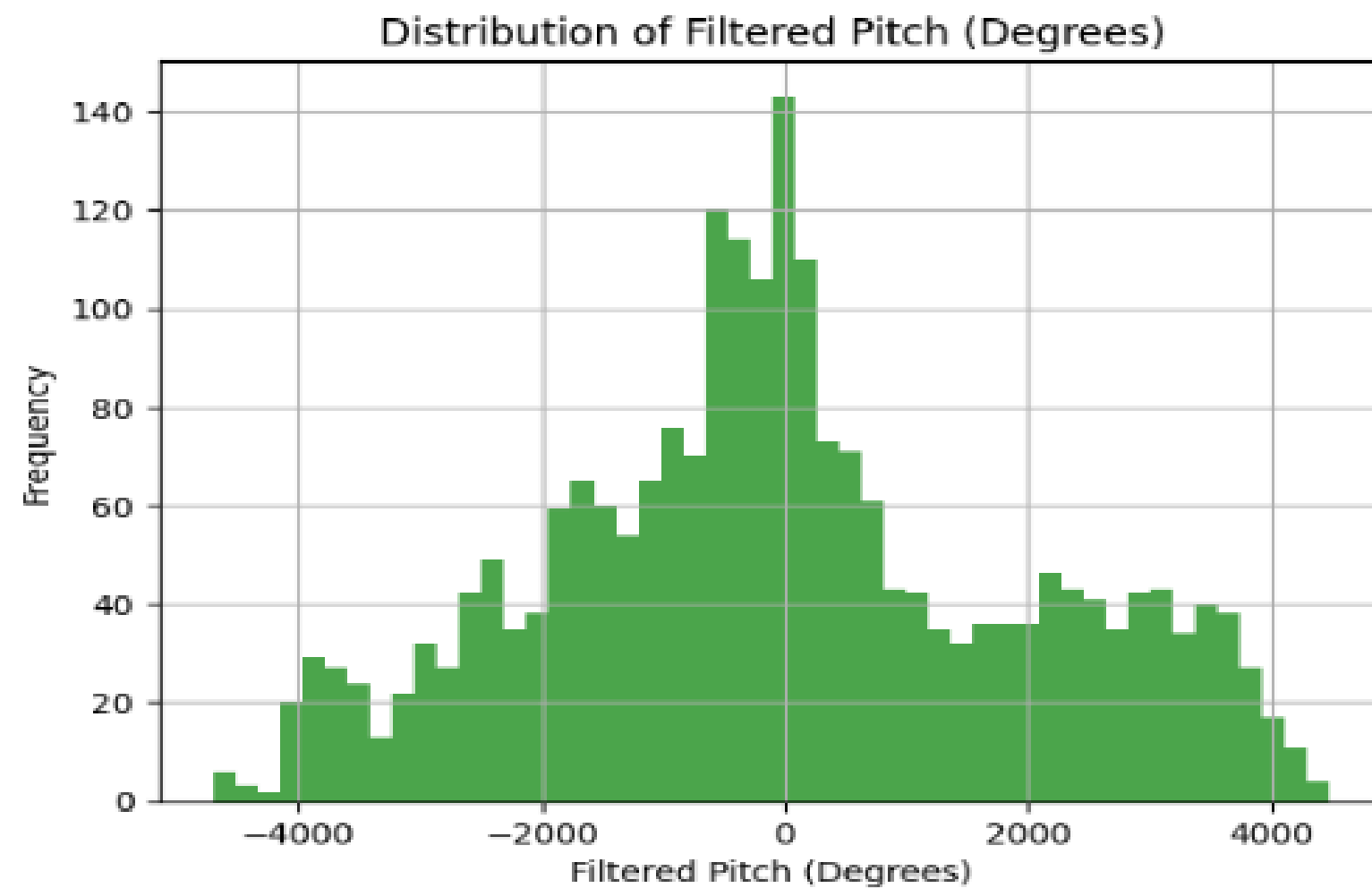
## Project Flow



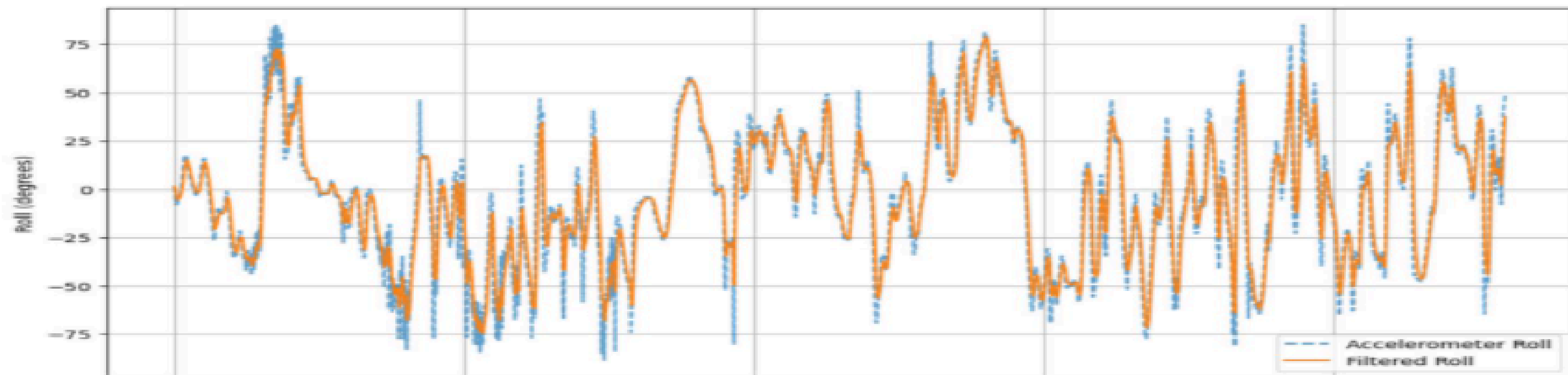
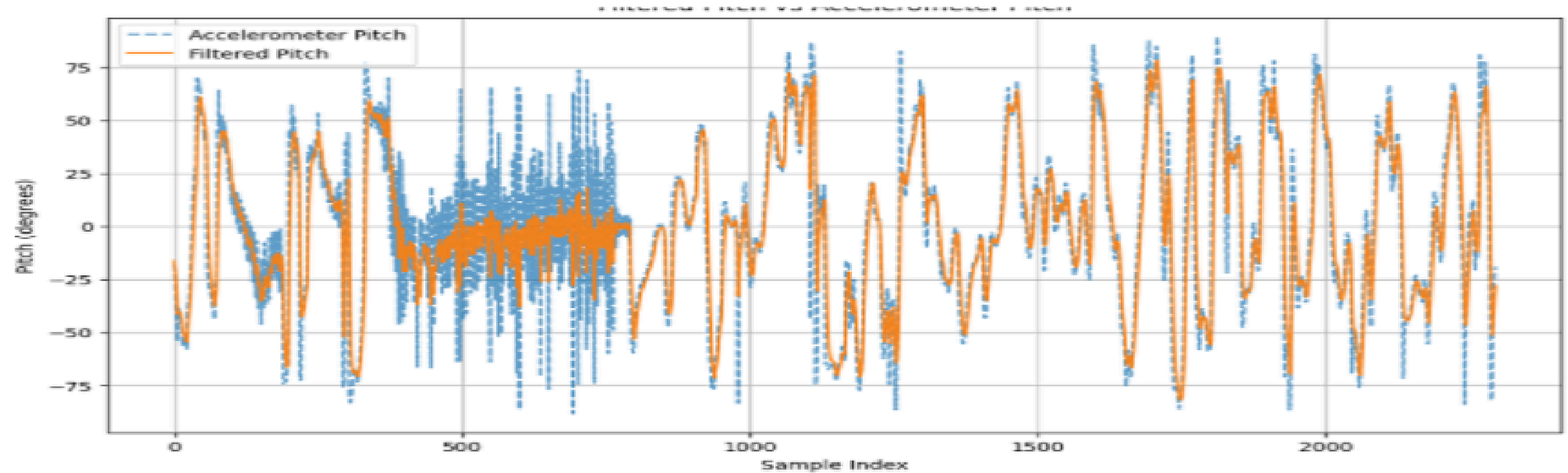
# Methodology

Feature Extraction:

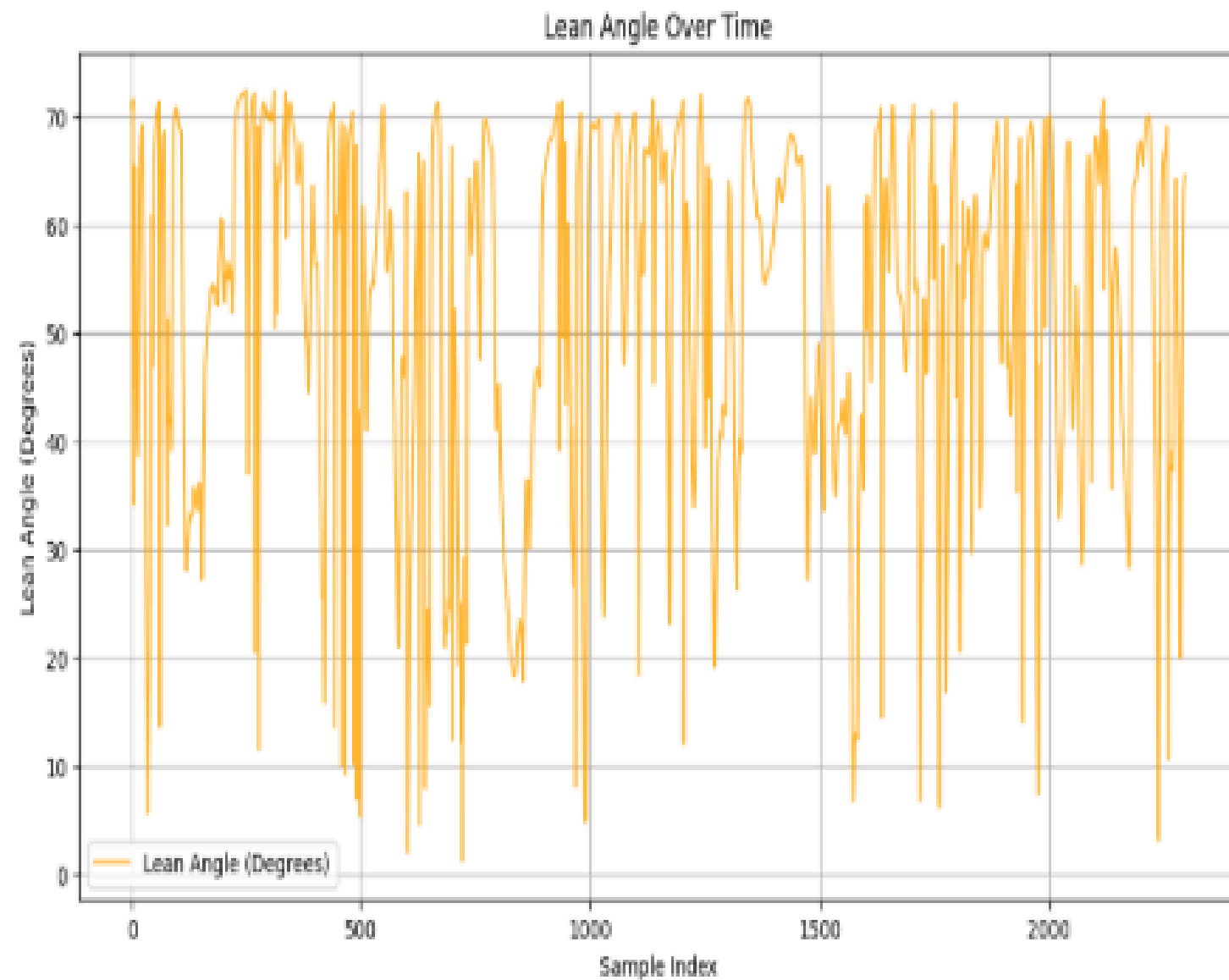
Compute physical metrics (pitch and roll) based on accelerometer data



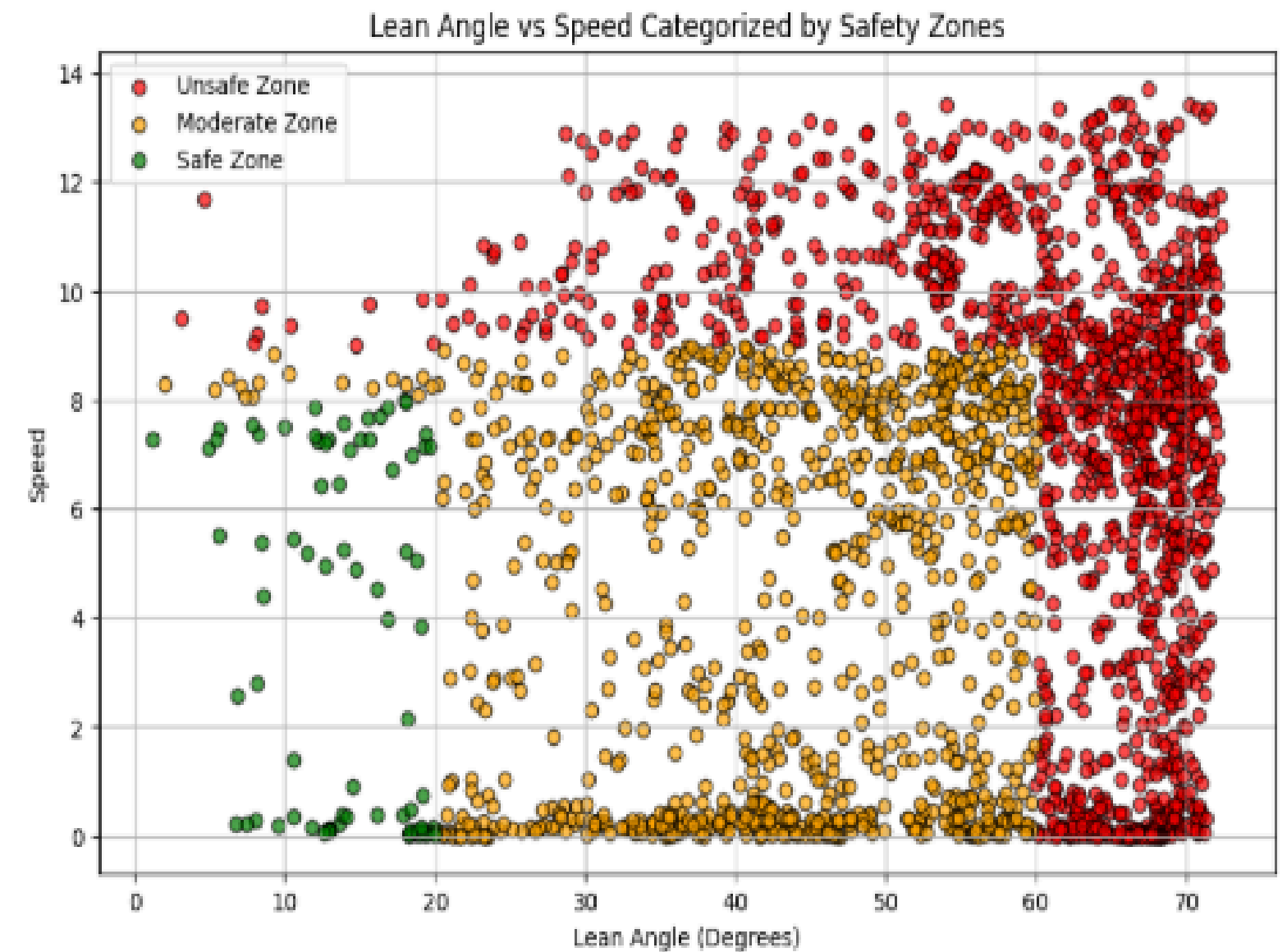
# Kalman Filter Implementation



# Lean Angle and Clustering



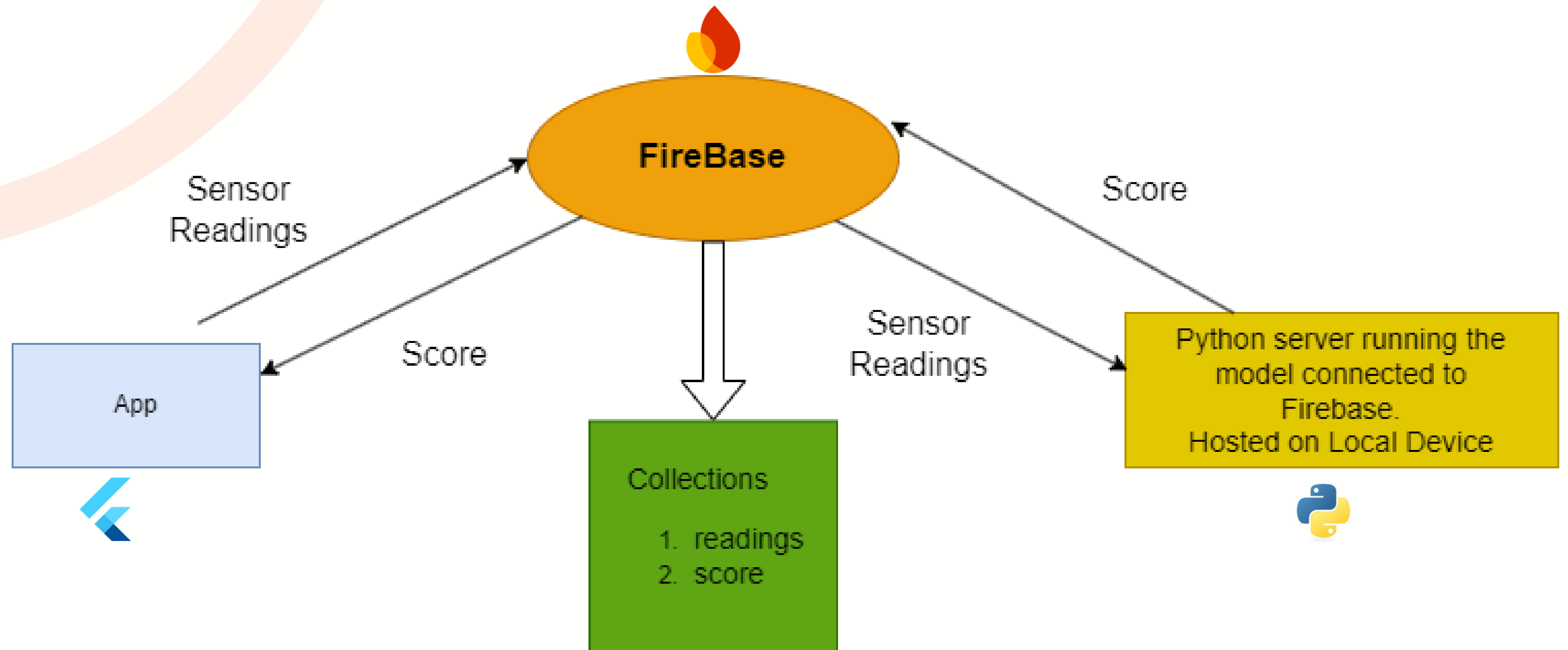
Cumulative Lean Angle



Clusters

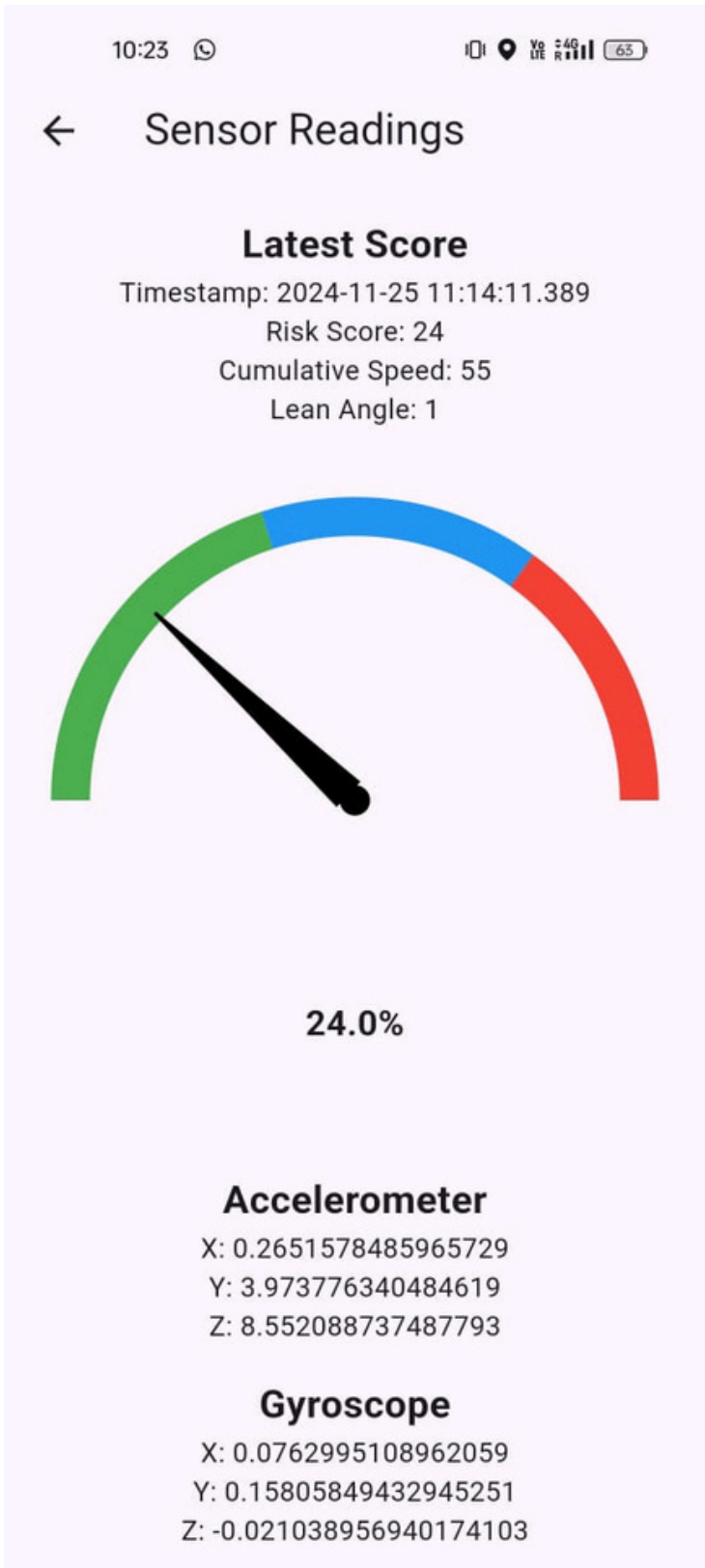


# Functioning of the App

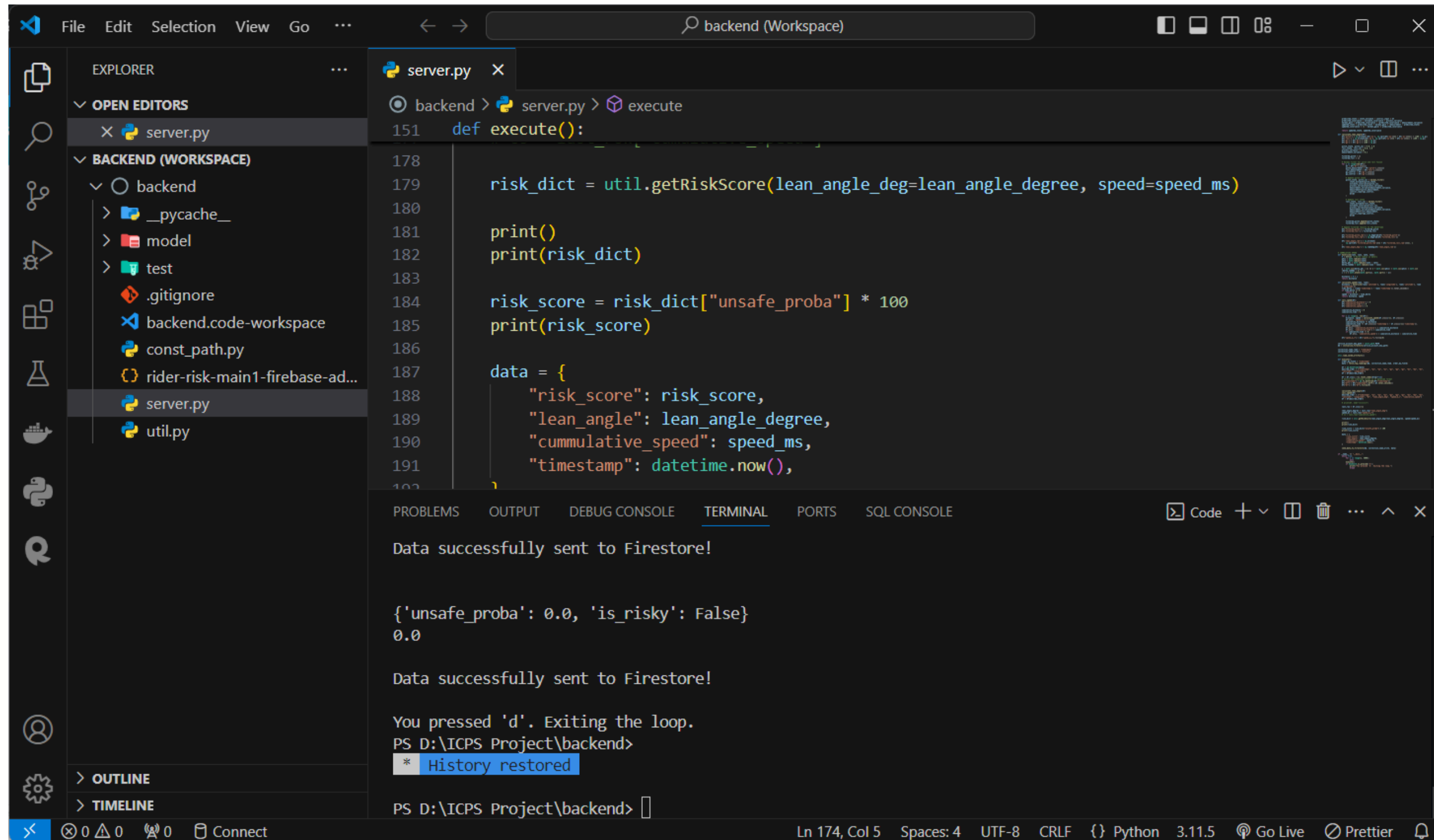




# Snapshots of the App



# Backend Preview - Python Server

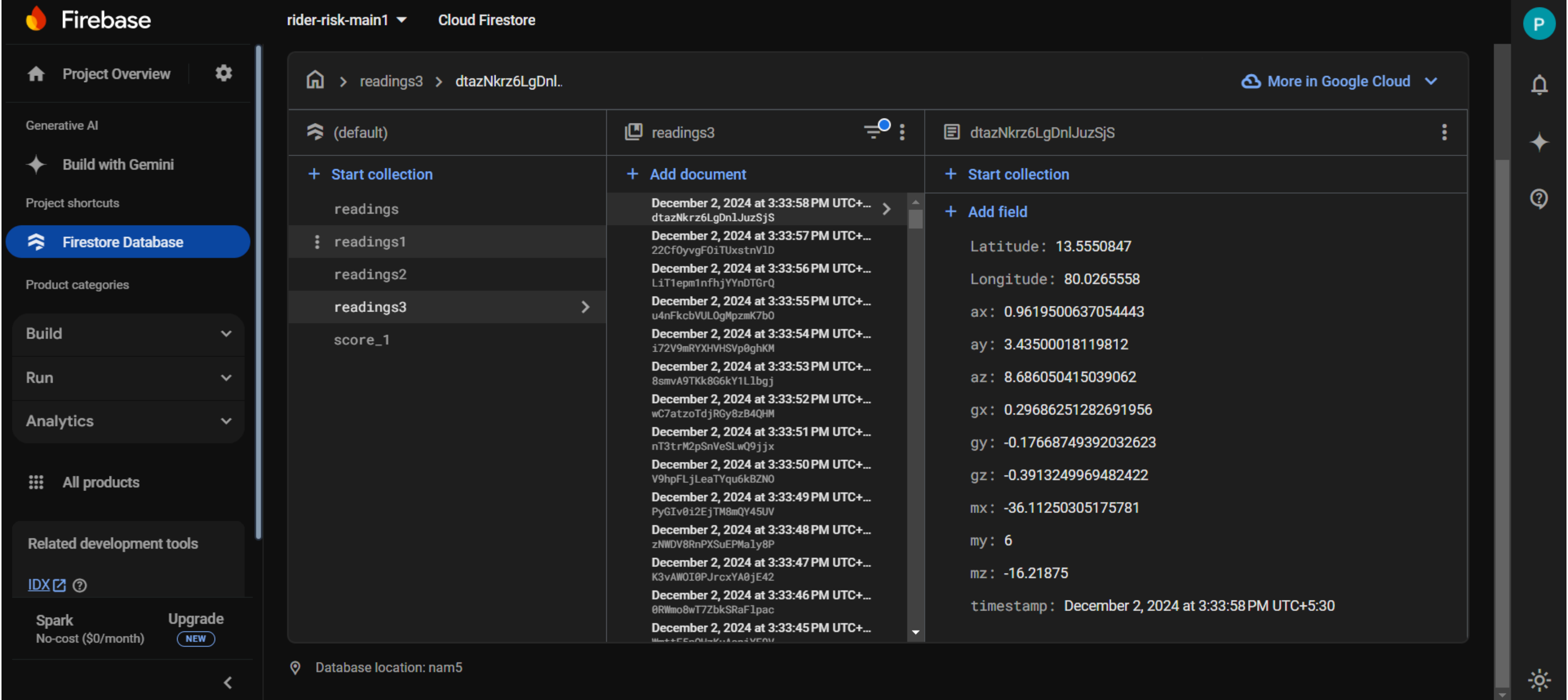


The image shows a Visual Studio Code editor window with a Python workspace named 'backend'. The Explorer sidebar on the left shows the project structure, including files like 'server.py' and 'util.py'. The main editor area displays the code for 'server.py', which defines an 'execute()' function. This function calls 'util.getRiskScore()' to calculate a risk score based on 'lean\_angle\_degree' and 'speed\_ms', prints the result, and then sends the data to Firestore. The bottom panel shows the terminal output, which confirms that the data was successfully sent to Firestore and displays the JSON response: {'unsafe\_proba': 0.0, 'is\_risky': False}. The terminal also shows the user pressing 'd' to exit the loop and the command prompt returning to 'PS D:\ICPS Project\backend>'.

```
File Edit Selection View Go ... backend (Workspace)
EXPLORER
  OPEN EDITORS
    server.py
  BACKEND (WORKSPACE)
    backend
      __pycache__
      model
      test
      .gitignore
      backend.code-workspace
      const_path.py
      rider-risk-main1-firebase-ad...
      server.py
      util.py
    OUTLINE
    TIMELINE

server.py
  backend > server.py > execute
  151 def execute():
  178
  179     risk_dict = util.getRiskScore(lean_angle_deg=lean_angle_degree, speed=speed_ms)
  180
  181     print()
  182     print(risk_dict)
  183
  184     risk_score = risk_dict["unsafe_proba"] * 100
  185     print(risk_score)
  186
  187     data = {
  188         "risk_score": risk_score,
  189         "lean_angle": lean_angle_degree,
  190         "cummulative_speed": speed_ms,
  191         "timestamp": datetime.now(),
  192     }
  193
  PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SQL CONSOLE
  Data successfully sent to Firestore!
  {'unsafe_proba': 0.0, 'is_risky': False}
  0.0
  Data successfully sent to Firestore!
  You pressed 'd'. Exiting the loop.
  PS D:\ICPS Project\backend>
  * History restored
  PS D:\ICPS Project\backend>
  Ln 174, Col 5 Spaces: 4 UTF-8 CRLF {} Python 3.11.5 Go Live Prettier
```

# Backend Preview - Firebase



The screenshot displays the Firebase Cloud Firestore console for the project 'rider-risk-main1'. The left sidebar shows the 'Project Overview' and 'Firestore Database' sections. The main area shows a collection named 'readings3' with a list of documents. The selected document, 'dtazNkrz6LgDn1JuzSjS', is shown in the right pane with its fields: Latitude, Longitude, ax, ay, az, gx, gy, gz, mx, my, mz, and timestamp.

**Firestore Database**

**readings3**

Document ID	Timestamp
dtazNkrz6LgDn1JuzSjS	December 2, 2024 at 3:33:58 PM UTC+...
22Cf0yvgF0iUxstnV1D	December 2, 2024 at 3:33:57 PM UTC+...
LiT1epm1nfhjYYnDTGrQ	December 2, 2024 at 3:33:56 PM UTC+...
u4nFkcbVUL0gMpzmk7b0	December 2, 2024 at 3:33:55 PM UTC+...
i72V9mRYXHVSvp0ghKM	December 2, 2024 at 3:33:54 PM UTC+...
8smvA9TKk8G6kY1L1bgj	December 2, 2024 at 3:33:53 PM UTC+...
wC7atzoTdjRGy8zB4QHM	December 2, 2024 at 3:33:52 PM UTC+...
nT3trM2pSnVeSLwQ9jjx	December 2, 2024 at 3:33:51 PM UTC+...
V9hpFLjLeaTYqu6kBZNO	December 2, 2024 at 3:33:50 PM UTC+...
PyGIv0i2EjTM8mQY45UV	December 2, 2024 at 3:33:49 PM UTC+...
zNWDV8RnPXSuEPMa1y8P	December 2, 2024 at 3:33:48 PM UTC+...
K3vAWOI0PJrcxYA0jE42	December 2, 2024 at 3:33:47 PM UTC+...
0RWmo8wT7ZbkSRaFlpac	December 2, 2024 at 3:33:46 PM UTC+...
WtttFEeQhKkApvYEDM	December 2, 2024 at 3:33:45 PM UTC+...

**dtazNkrz6LgDn1JuzSjS**

Field	Value
Latitude	13.5550847
Longitude	80.0265558
ax	0.9619500637054443
ay	3.43500018119812
az	8.686050415039062
gx	0.29686251282691956
gy	-0.17668749392032623
gz	-0.3913249969482422
mx	-36.11250305175781
my	6
mz	-16.21875
timestamp	December 2, 2024 at 3:33:58 PM UTC+5:30

Database location: nam5



# Challenges

- Real-Time Data Accuracy
- Variability in Riding Styles
- Dynamic Environmental Factors



# Future Plans

- Weather-Integrated Risk Adjustment
- Terrain and Elevation Analysis
- Traffic-Aware Risk Assessment
- Insurance companies using it to determine premiums

