NASA SPACE APPS NOIDA

NASA Space Apps Noida 2024

World's Largest Space & Science Hackathon

5-6th October 2024 | 36 Hours Hackathon

Innovation partner H2S

TEAM DETAILS

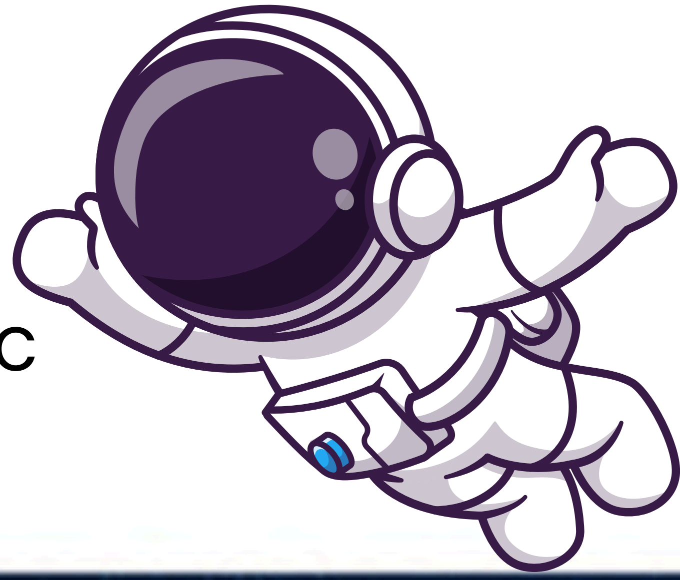
TEAM NAME: COSMIC CHAKRA
TEAM LEADER NAME: MALAVIKA GUPTA
PROBLEM STATEMENT: SEISMIC DETECTION ACROSS THE SOLAR SYSTEM

Brief about the idea

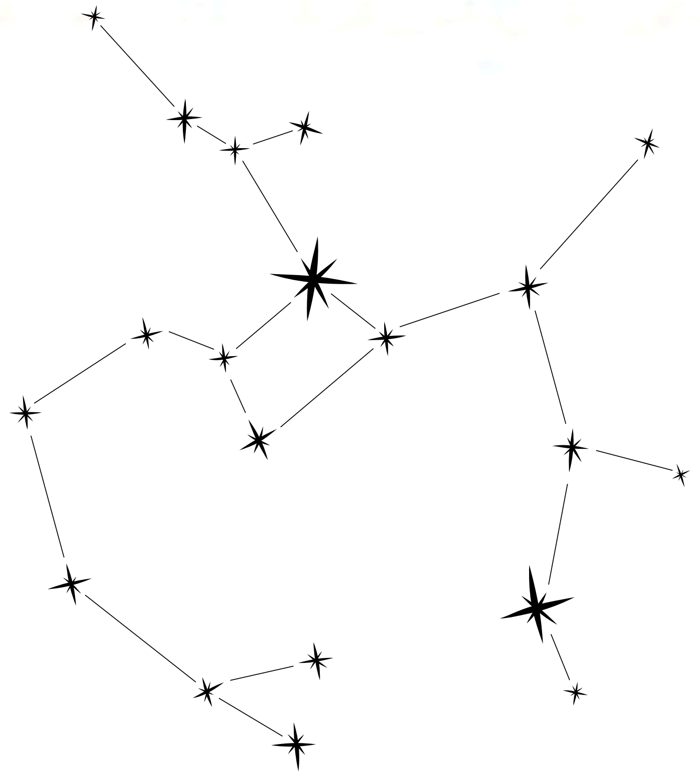
Imagine being able to eavesdrop on the heartbeat of a planet. That's essentially what planetary seismology does. Missions like Apollo on the Moon and InSight on Mars have planted high-tech ears on these distant worlds, listening intently to their geological whispers and rumbles.

The challenge? These cosmic microphones pick up everything - from the faintest tremors to the loudest quakes, and even the background noise of the lander itself. It's like trying to hear a pin drop in a busy café. Sending all this data back to Earth is like trying to email a library's worth of books using a dial-up connection - it's slow and drains the lander's precious energy.

Here's where our idea comes in: What if we could teach the lander to be a smart listener? We want to develop a program that acts like a audio engineer, identifying the 'hit singles' (significant quakes) from all the background noise. By developing a program capable of identifying seismic quakes amidst noise directly on the lander, we can optimize the transmission by only sending useful data. This will conserve energy and increase the efficiency of data utilization, ultimately enhancing scientific research on planetary seismic activity.



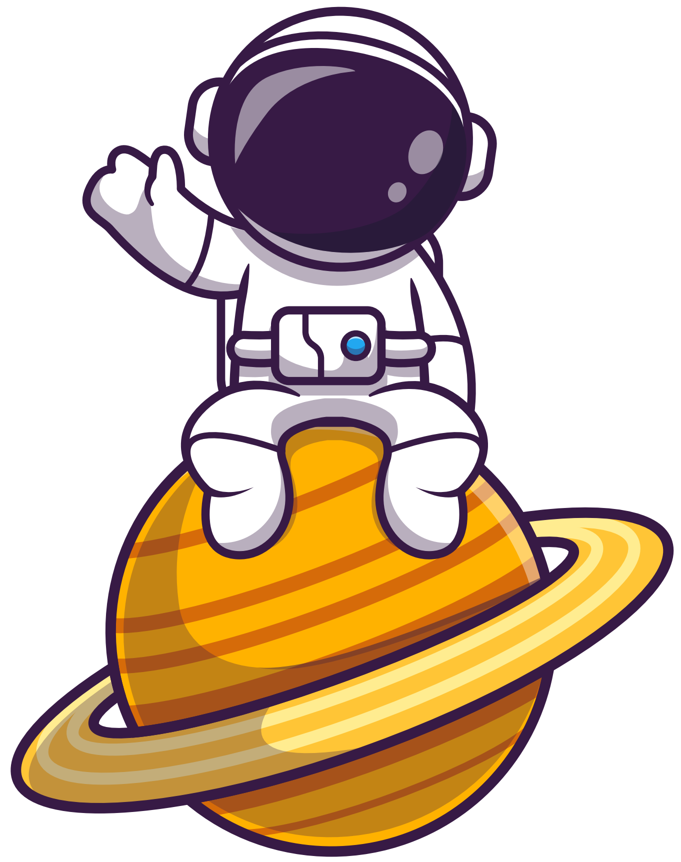
Opportunities



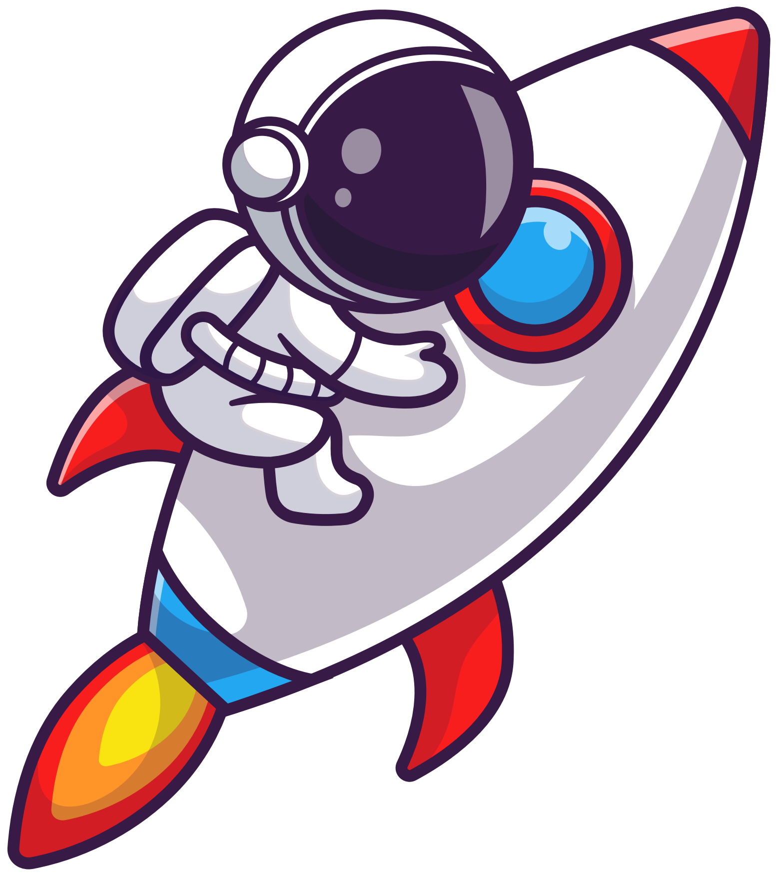
How Different is it from Other Existing Ideas? This solution implements on-board, real-time seismic data analysis, unlike current systems that transmit all collected data indiscriminately. By filtering and prioritizing data at the source, it dramatically reduces transmission volume and energy consumption.

How Will it Solve the Problem? The system addresses the energy constraints of planetary missions by significantly decreasing data transmission requirements. This conservation of power extends the operational lifespan of the lander, enabling the collection of more valuable seismic data over a longer period.

Unique Selling Point (USP): The core innovation lies in the autonomous, intelligent discrimination between significant seismic events and background noise directly on the lander. This smart filtering approach optimizes data transmission, maximizes the scientific value of the mission, and substantially improves the efficiency of limited power resources.

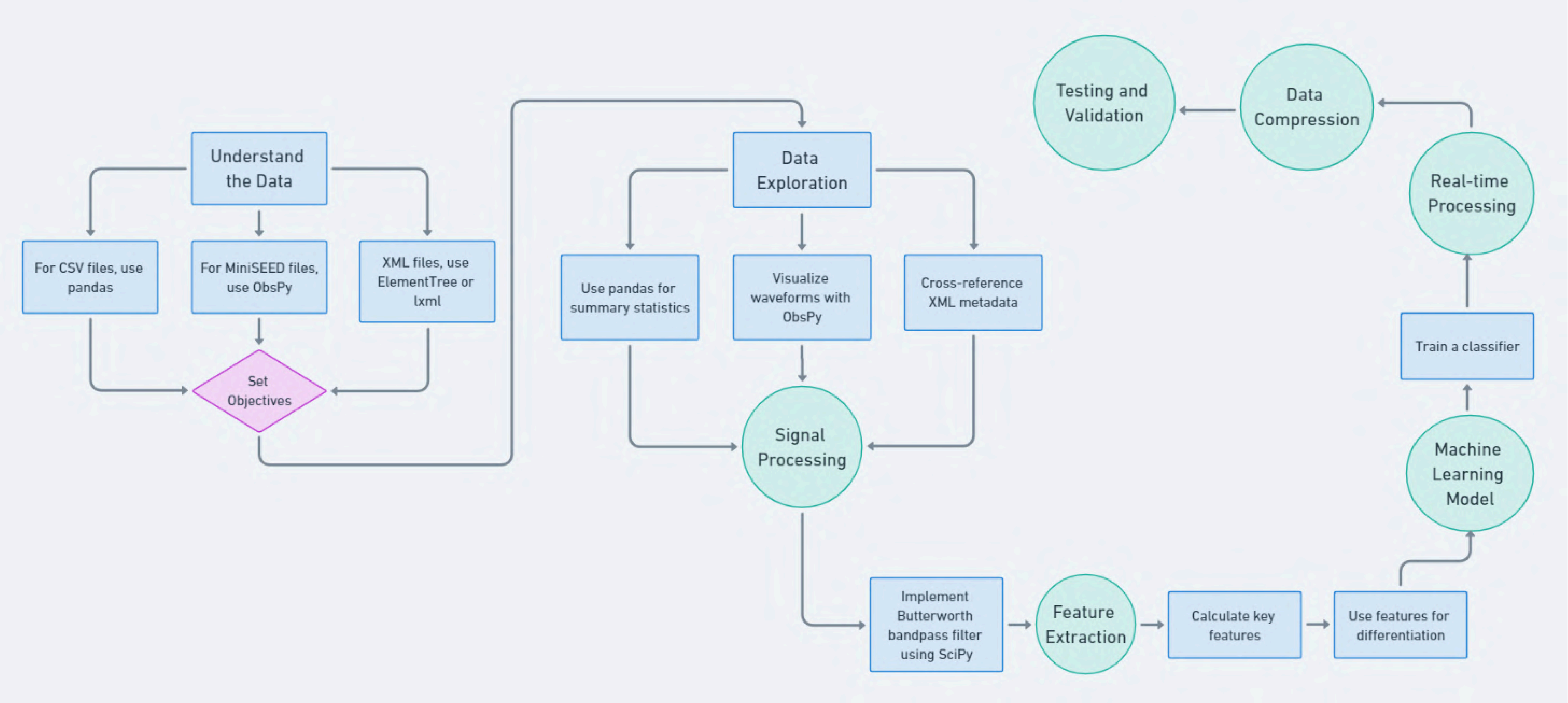


List of features offered by the solution

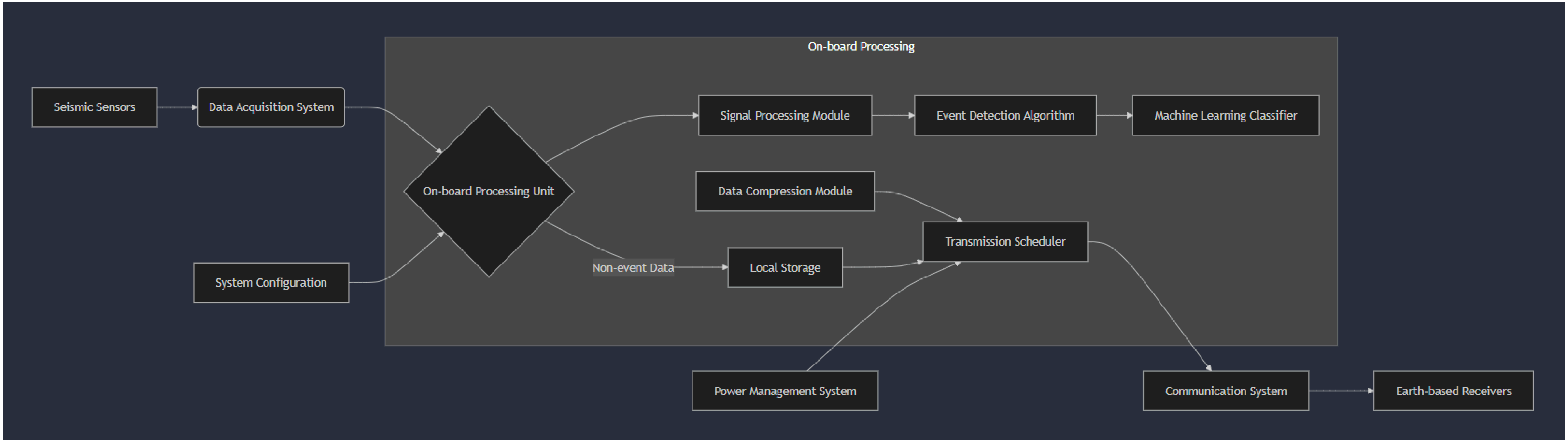


- Real-time seismic event detection and noise filtering algorithms
- Adaptive data compression and prioritization system
- Energy-saving mechanism by reducing unnecessary transmissions
- Seamless integration with existing planetary lander hardware
- Machine learning component for continuous improvement in detection accuracy
- Compatibility with various seismometer types and configurations
- User-friendly interface for remote calibration and parameter adjustment
- Scalable model for future planetary seismic missions

Process flow diagram



Architecture diagram of the proposed solution



Technologies used in the solution

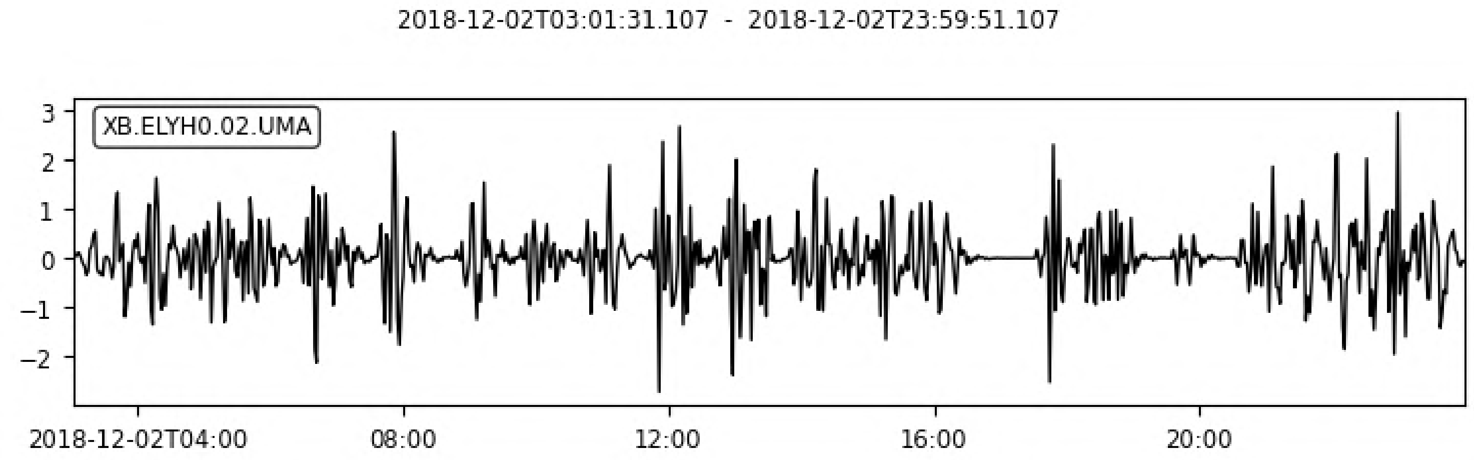
- Python for data processing and algorithm implementation
- NumPy and SciPy for numerical computations and signal processing
- ObsPy for seismological data analysis and MiniSEED file handling
- Pandas for CSV data manipulation and analysis
- Scikit-learn for implementing machine learning models
- TensorFlow or PyTorch for deep learning approaches
- Custom data compression algorithms optimized for seismic waveforms
- UDP-based protocols for reliable low-power data transmission
- XML parsing libraries (e.g., ElementTree) for metadata handling
- Git for version control and collaborative development

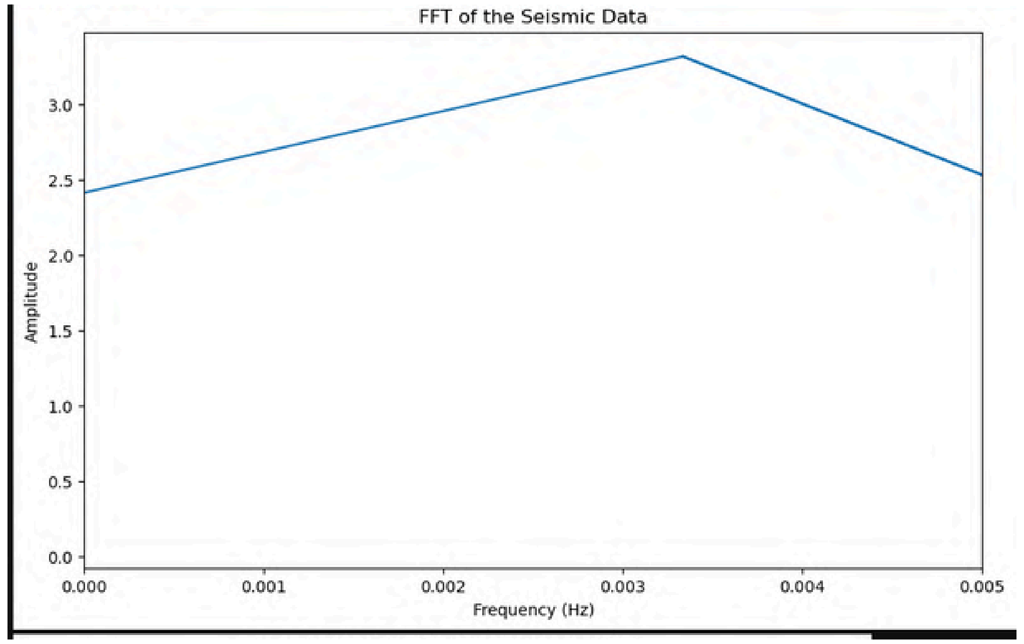
NASA Space Apps Noida 2024

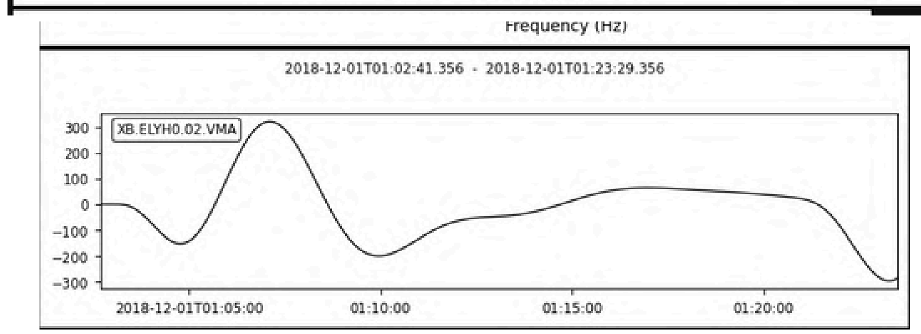
Innovation partner

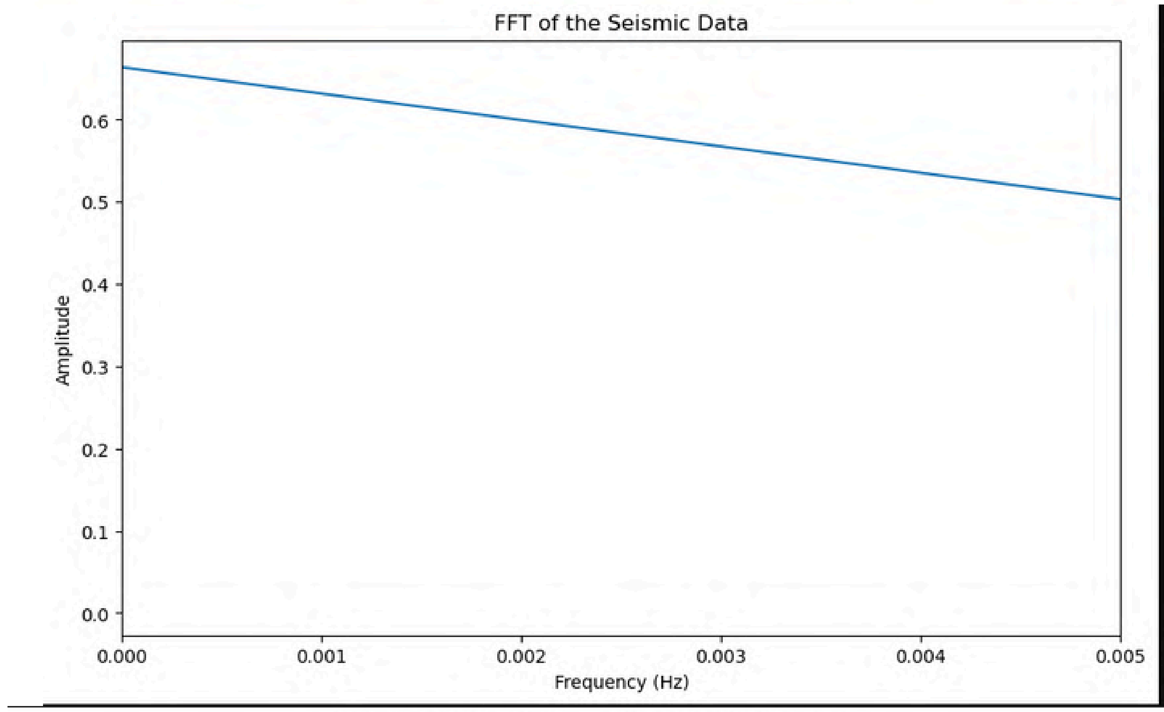
```python
# Preprocess function
def preprocess_stream(stream, filter_min_freq, filter_max_freq, taper_pct=0.05):
    try:
        for trace in stream:
            # Basic preprocessing steps with error handling
            trace.detrend(type='linear')
            trace.taper(max_percentage=taper_pct, type='cosine')
            trace.filter('bandpass', freqmin=filter_min_freq, freqmax=filter_max_freq)
        return stream
    except Exception as e:
        print(f"Error during preprocessing: {e}")
        return None  # Return None if preprocessing fails
```

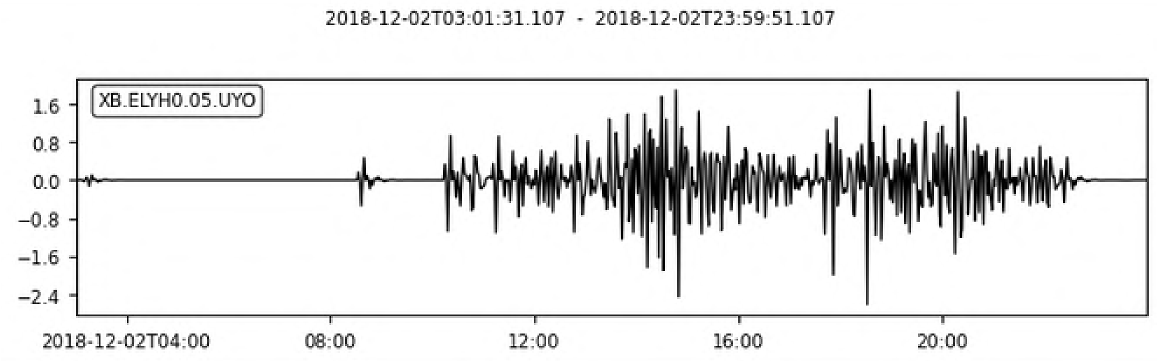
Prototype

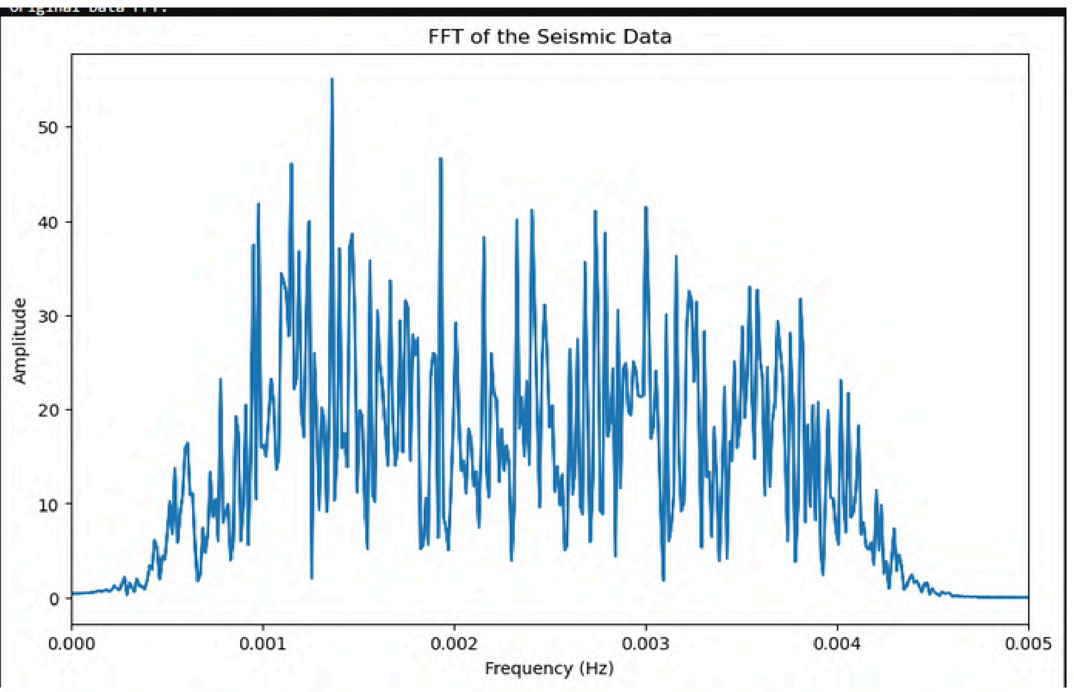


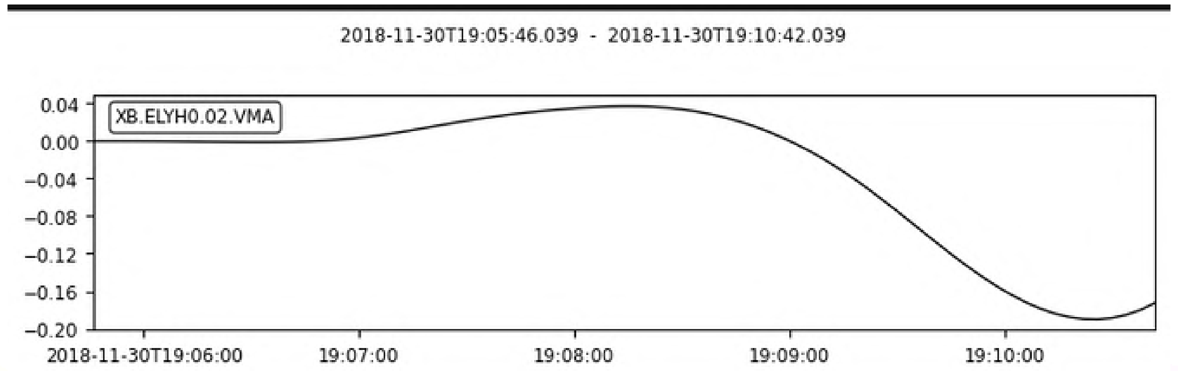


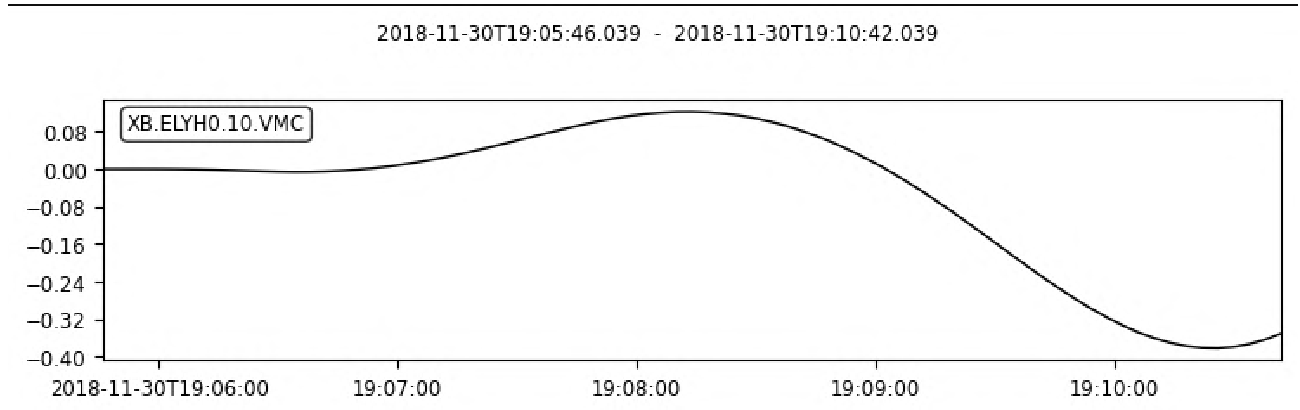










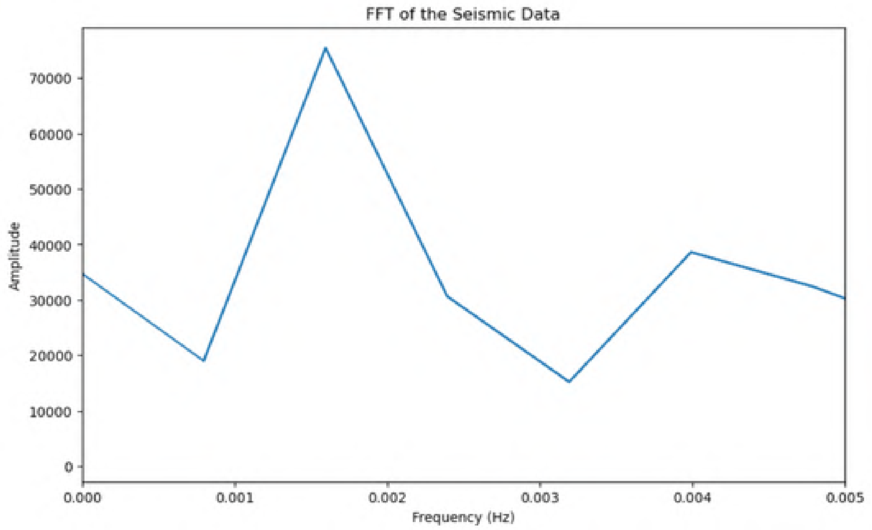


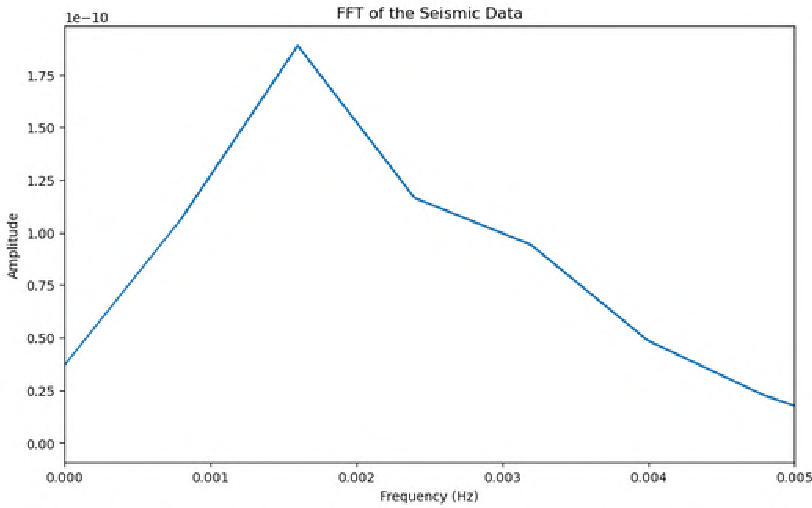
NASA Space Apps Noida 2024

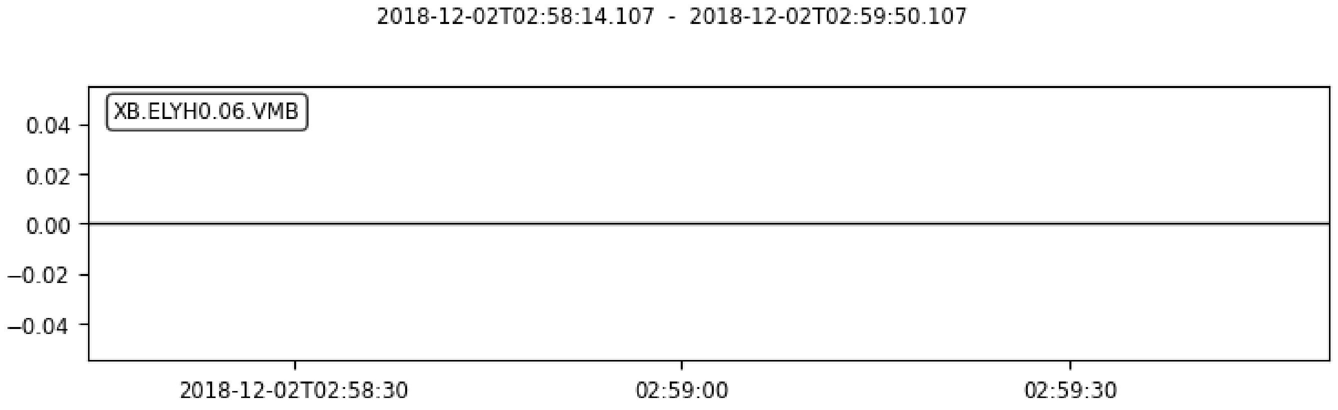
NASA SPACE APPS NOIDA

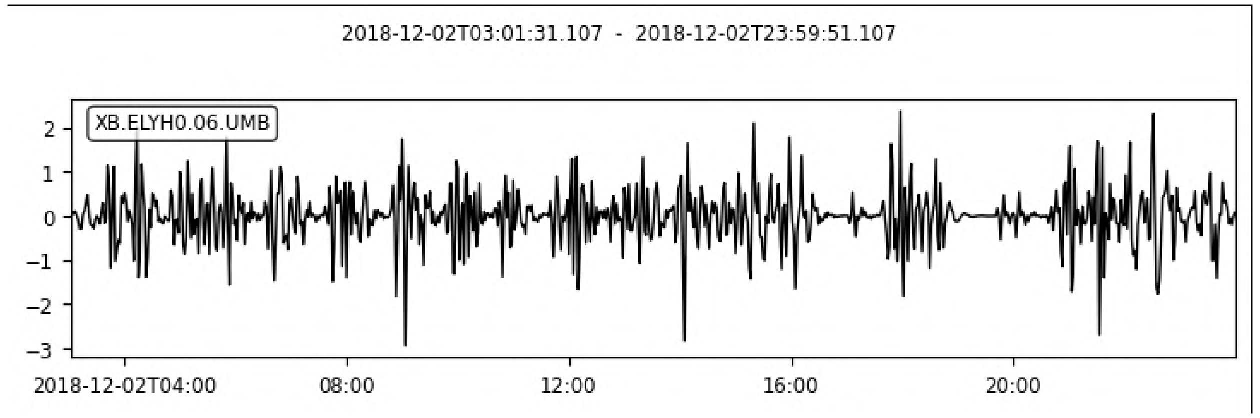
Innovation partner H2S HACK2SKILL

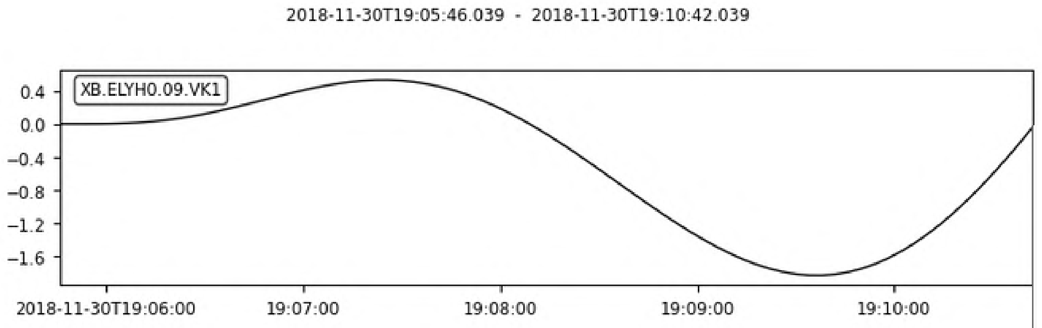
```python
def extract_features(trace, window_size=50, step_size=5):
    feature_matrix = []
    data = trace.data

    if len(data) == 0:
        print("Empty trace data, skipping...")
        return np.array([])

    try:
        # Calculate the number of windows
        num_windows = (len(data) - window_size) // step_size + 1
        print(f"Number of windows: {num_windows}")

        for start in range(0, len(data) - window_size + 1, step_size):
            end = start + window_size
            window = data[start:end]

            if len(window) < window_size:
                print(f"Skipping window due to insufficient data: {len(window)}")
                continue

            # Existing and new feature calculations
            max_amp = np.max(window)
            min_amp = np.min(window)
            mean_amp = np.mean(window)
            skewness = skew(window, bias=False)
            kurt = kurtosis(window, bias=False)
            # Frequency domain features
            freq_data = np.abs(fft(window))
            mean_freq = np.mean(freq_data)
            max_freq = np.max(freq_data)
            peak_freq = np.argmax(freq_data)
            freq_variance = np.var(freq_data)

            # Additional Features
            zero_crossings = np.where(np.diff(np.sign(window)))[0]
            zero_crossing_rate = len(zero_crossings) / len(window)

            power_spectrum = np.abs(freq_data) ** 2
            power_spectrum = power_spectrum / np.sum(power_spectrum)  #
            spectral_entropy = entropy(power_spectrum)

            # Updated feature vector
            feature_vector = [
                max_amp, min_amp, mean_amp, skewness, kurt,
                mean_freq, max_freq, peak_freq, freq_variance,
                zero_crossing_rate, spectral_entropy
            ]
            feature_matrix.append(feature_vector)

    except Exception as e:
        print(f"Error during feature extraction: {e}")
```

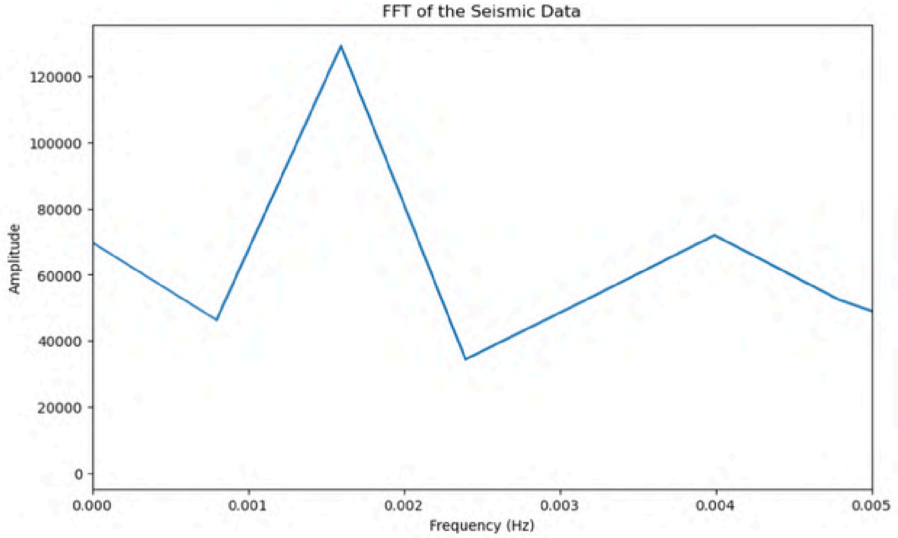


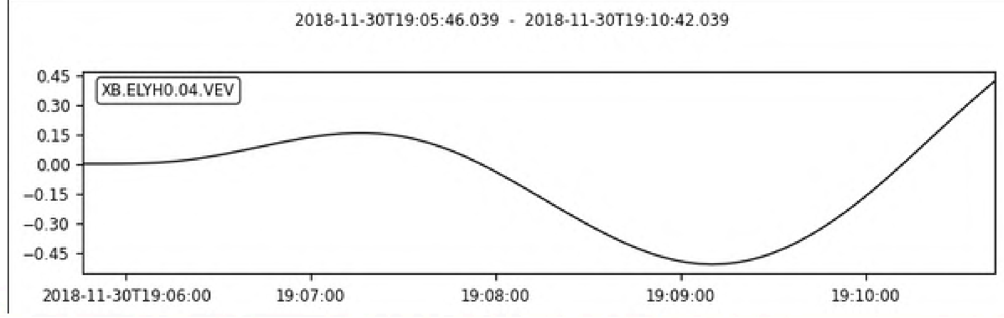


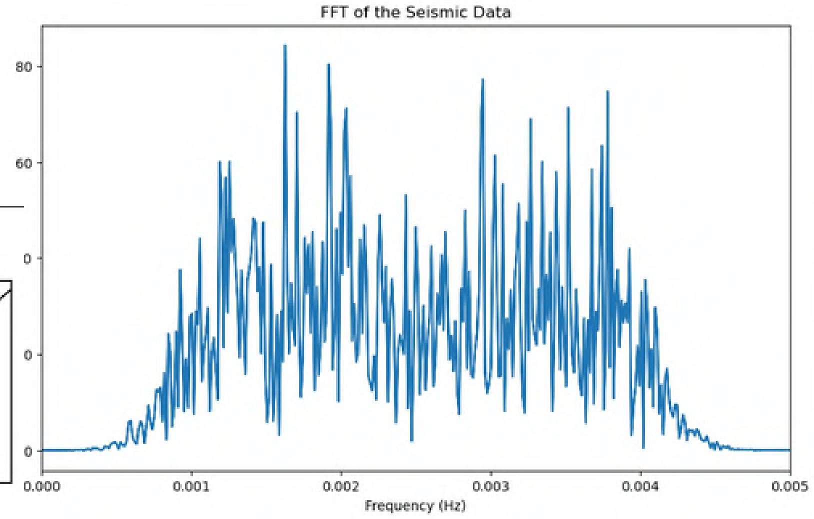








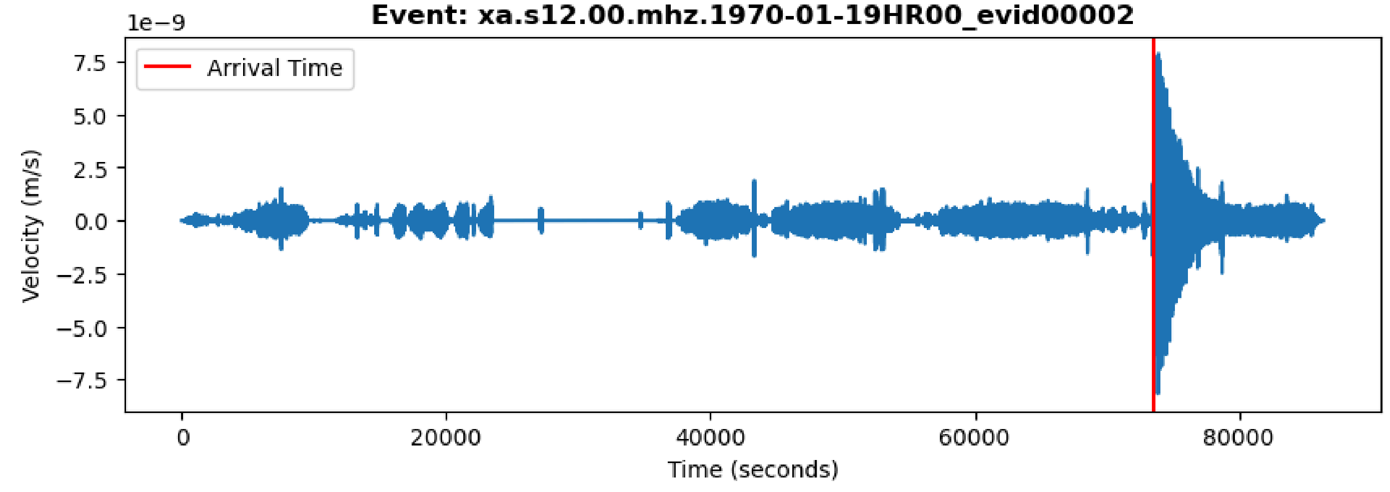




NASA Space Apps Noida 2024

Innovation partner

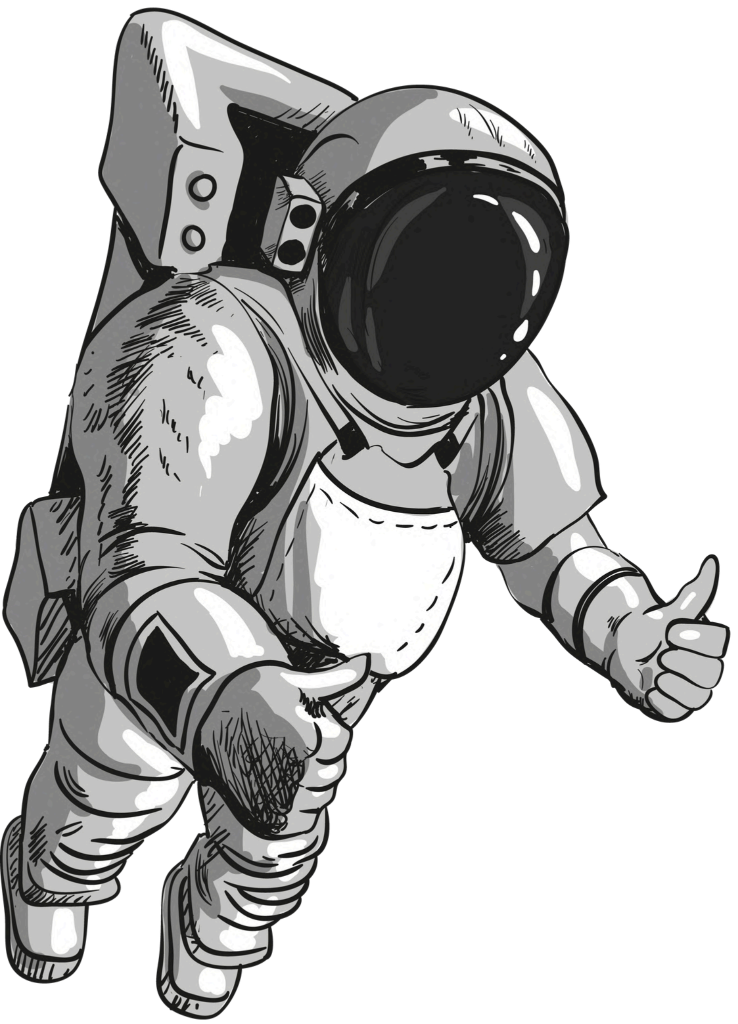
```python
# Function to preprocess and filter seismic data
def preprocess_and_filter(trace, filter_min_freq=0.001, filter_max_freq=0.004, taper_pct=0.05):
    trace.detrend(type='linear')
    trace.taper(max_percentage=taper_pct, type='cosine')
    trace.filter('bandpass', freqmin=filter_min_freq, freqmax=filter_max_freq)
    return trace
```




```python
# Function to extract features from the seismic event window
def extract_features_from_trace(trace, window_size=50, step_size=5):
    feature_matrix = []
    data = trace.data

    num_windows = (len(data) - window_size) // step_size + 1
    for start in range(0, len(data) - window_size + 1, step_size):
        end = start + window_size
        window = data[start:end]
        max_amp = np.max(window)
        min_amp = np.min(window)
        mean_amp = np.mean(window)
        window_skewness = skew(window, bias=False)
        window_kurtosis = kurtosis(window, bias=False)
        freq_data = np.abs(fft(window))
        mean_freq = np.mean(freq_data)
        max_freq = np.max(freq_data)
        peak_freq = np.argmax(freq_data)
        freq_variance = np.var(freq_data)

        feature_vector = [max_amp, min_amp, mean_amp, window_skewness, window_kurtosis, mean_freq, max_freq, peak_freq, fre
        feature_matrix.append(feature_vector)

    return np.array(feature_matrix)
```

```python
# Example: Preprocess and extract features from the first event
file_path = f'{data_directory}{row["filename"]}.mseed'
stream = obspy.read(file_path)

# Apply preprocessing and filtering to the trace
for trace in stream:
    preprocessed_trace = preprocess_and_filter(trace)

    # Extract features
    features = extract_features_from_trace(preprocessed_trace)
    print(f"Extracted Features:\n{features}")

# Function to label traces based on a threshold
def label_trace(trace, threshold=1000):
    return 1 if np.max(trace.data) > threshold else 0

# Example: Label the preprocessed trace
label = label_trace(preprocessed_trace)
print(f"Label for the event: {label}")
```

Additional Details/Future Development



- Develop machine learning models capable of adjusting to different planetary conditions (e.g., higher seismic activity on Io, or the unique atmospheric interference on Venus).
- Implement transfer learning techniques to quickly adapt existing models to new planetary environments.
- Create interfaces to correlate seismic data with other planetary science measurements (e.g., atmospheric data, magnetic field readings) for more comprehensive analysis.
- Develop multi-modal event detection algorithms that combine data from various instruments to improve accuracy.
- Implement dynamic power allocation based on seismic activity levels and available energy resources.
- Research and implement lossy compression algorithms specifically tailored for seismic data, preserving key features while drastically reducing data volume.

GitHub Public Repository:

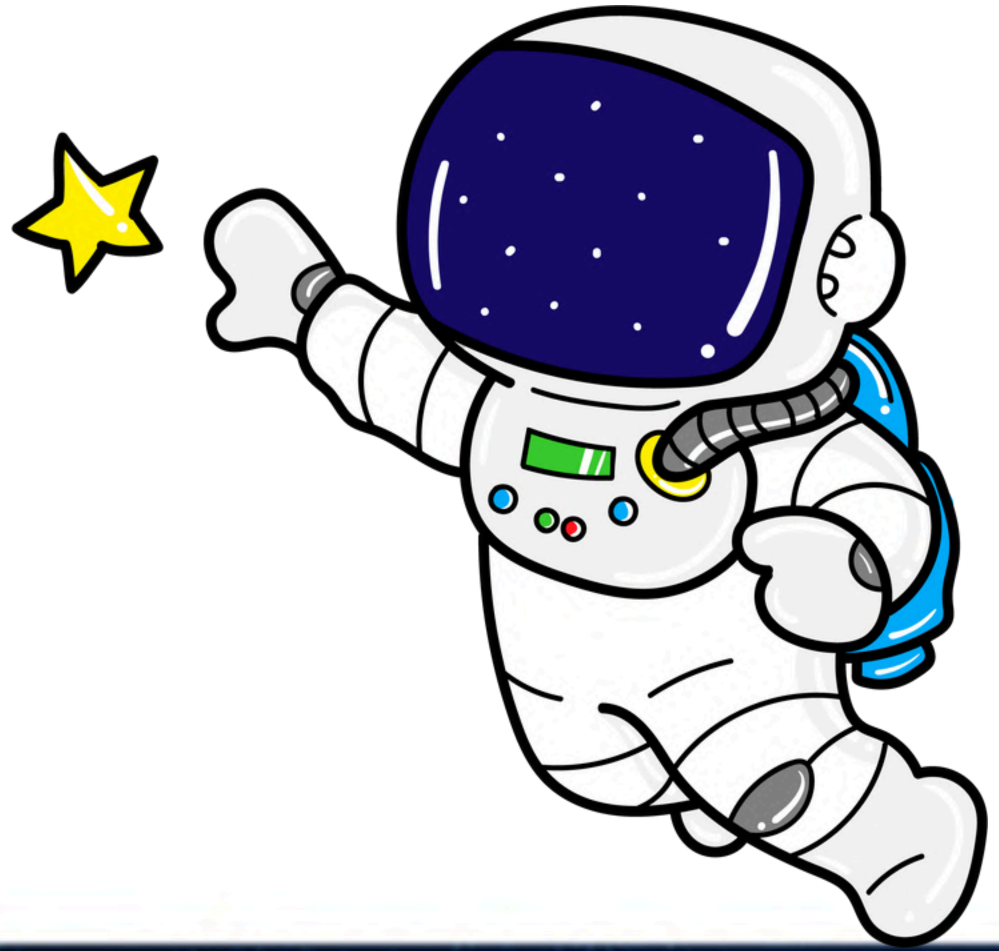
https://github.com/Malavika-Gupta/Optimized-Seismic-Data-Transmission-NASA-Noida-Space-Apps-2024-

Current Progress

- Utilizing Mars InSight MiniSEED data and Lunar data as provided by NASA
- Implemented data preprocessing and feature extraction
- Trained Random Forest classifier for event detection
- Achieved initial 100% accuracy

Next Steps

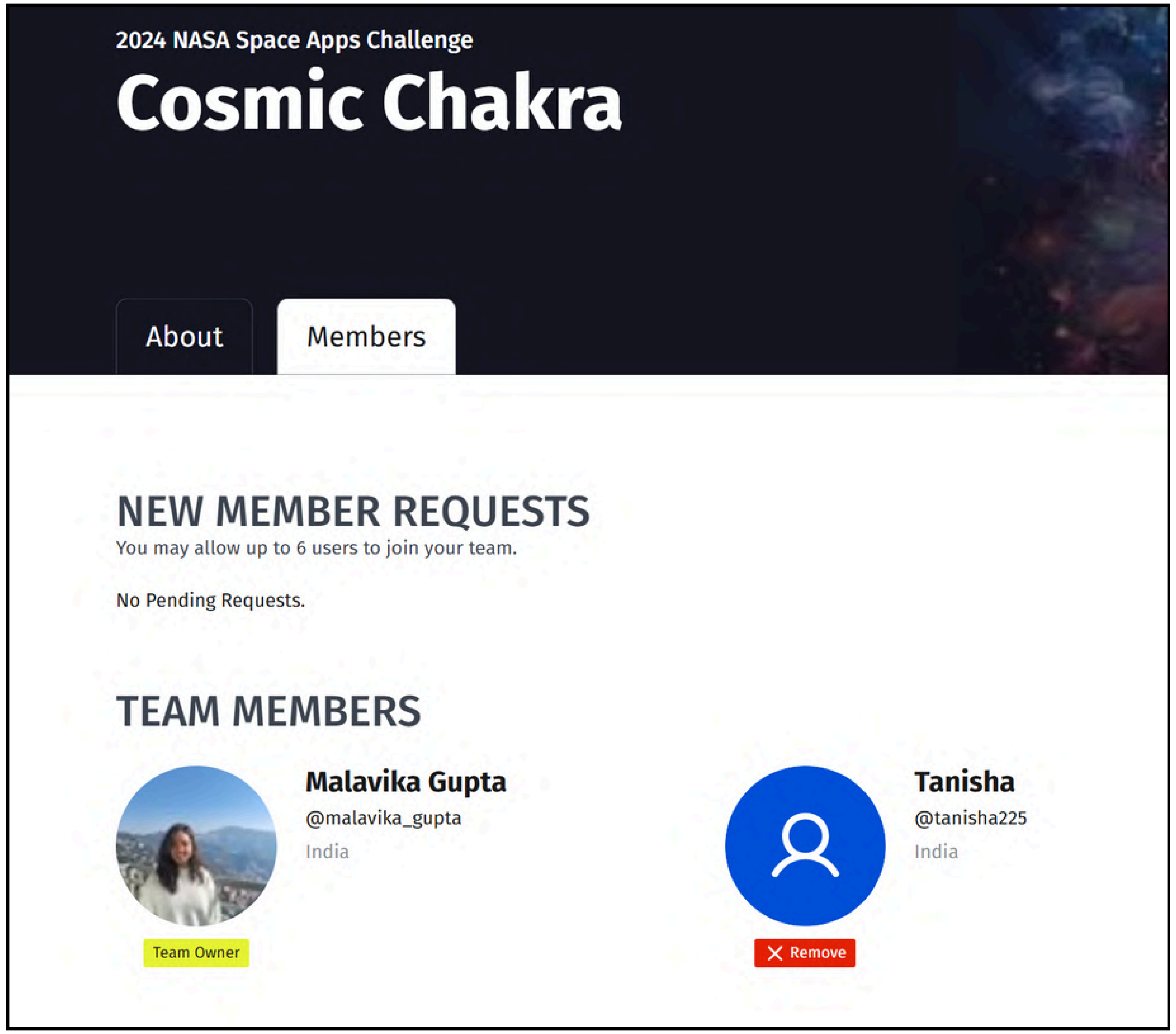
- Develop real-time processing simulation
- Integrate data compression algorithms
- Design on-board system architecture

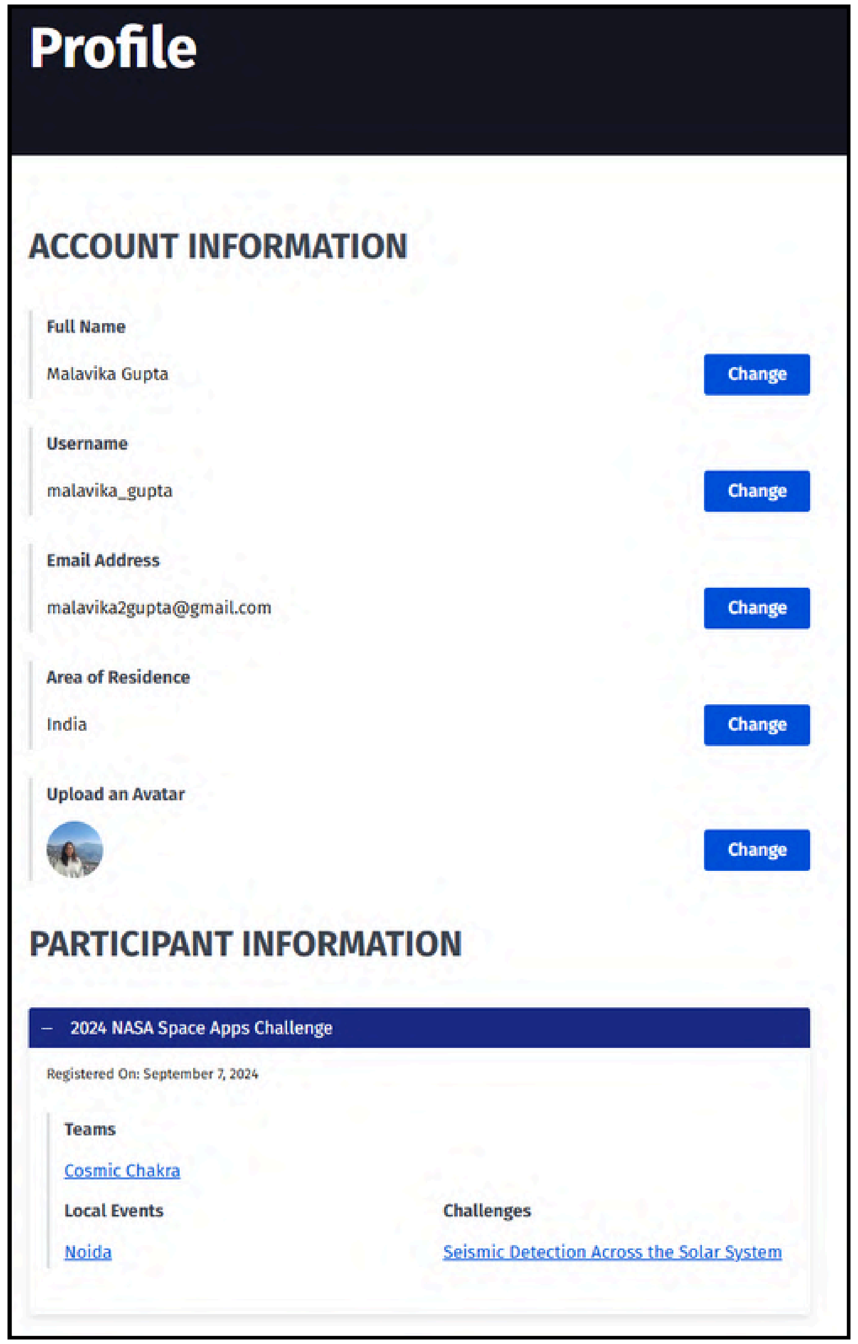


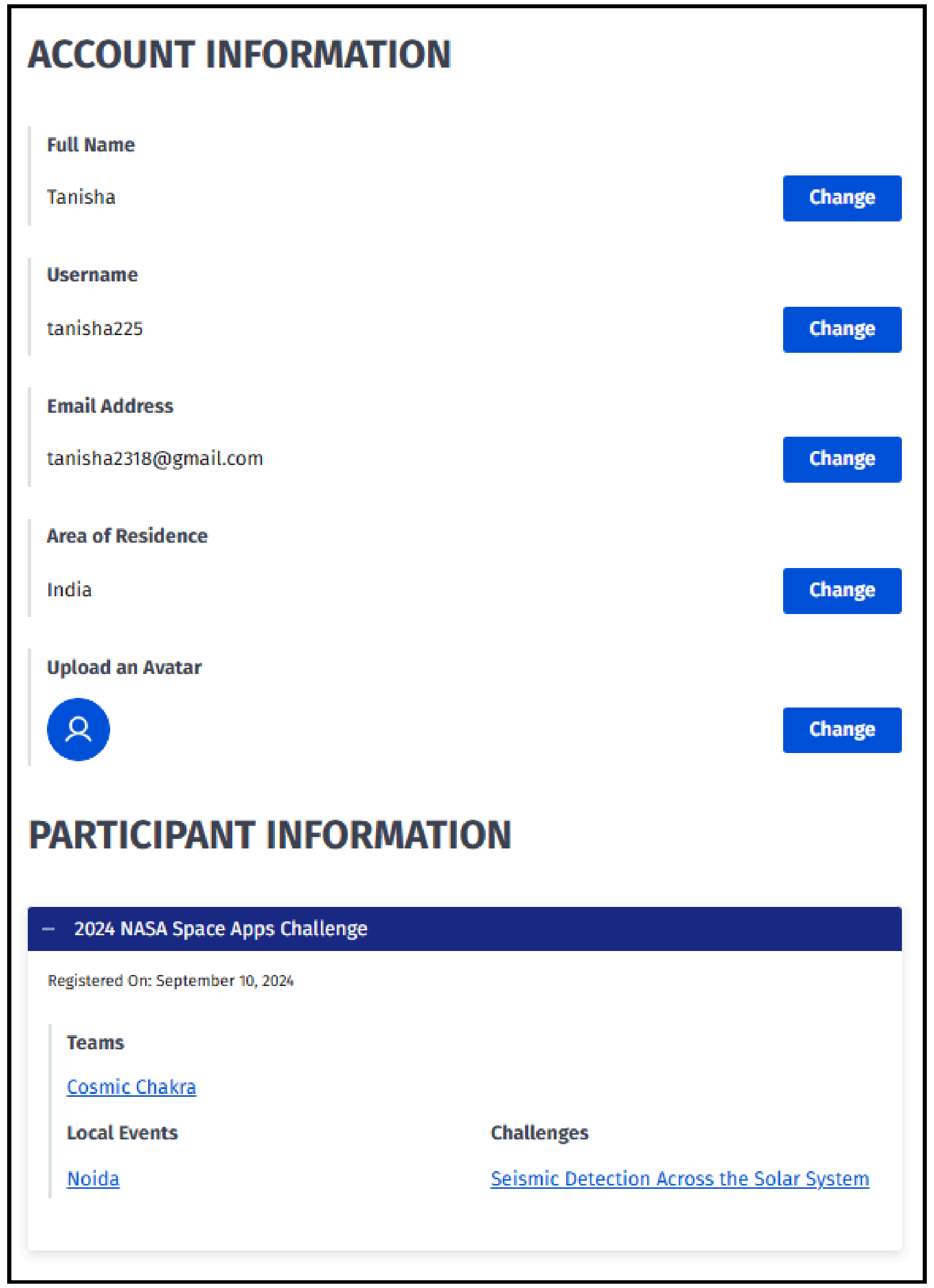
NASA Space Apps Noida 2024

Innovation partner

Proof of Registration on - https://www.spaceappschallenge.org/nasa-space-apps-2024/2024-local-events/noida







NASA SPACE APPS NOIDA | Innovation partner H2S HACK2SKILL

NASA Space Apps Noida 2024

World's Largest Space & Science Hackathon

Thank You