Designing a platform to receive and display real-time temperature and humidity data from IoT devices for environmental monitoring involves several key components. Below is an outline of the architecture and technologies you can use to create such a platform:

1. IoT Devices:

- Choose IoT devices that are capable of measuring temperature and humidity, such as sensors like DHT22, DHT11, or specialized IoT modules (e.g., ESP8266, ESP32).

- Ensure the devices are connected to the internet, either through Wi-Fi, cellular, or other appropriate connectivity options.

2. Data Collection and Transmission:

- IoT devices should send data in real-time to a central server. MQTT (Message Queuing Telemetry Transport) is a common protocol for this purpose due to its lightweight and efficient nature.

- Secure the data transmission using protocols like HTTPS, TLS/SSL, or other encryption methods to protect data integrity and privacy.

3. Cloud Server/Backend:

- Implement a cloud-based server or backend system that receives data from the IoT devices.

- Use a serverless architecture (e.g., AWS Lambda, Google Cloud Functions) to handle incoming data and reduce operational overhead.

- Store the incoming data in a database for historical reference, such as AWS DynamoDB, Google Cloud Firestore, or a relational database like PostgreSQL.

4. Authentication and Security:

- Implement user authentication to restrict access to the data. OAuth2 or API keys are common methods for securing API access.

- Secure your server and IoT devices with proper access control and secure key management.

5. Real-Time Data Processing:

- Implement real-time data processing to analyze and validate incoming data. This can include data normalization and outlier detection.

- Use stream processing technologies like Apache Kafka, Apache Flink, or AWS Kinesis for real-time data processing.

6. APIs:

- Create RESTful or GraphQL APIs for data retrieval and manipulation. These APIs will allow users to access the data in real-time.

- Use API gateways like AWS API Gateway or API management platforms like Apigee to manage and secure your APIs.

7. Data Storage and Databases:

- Choose an appropriate database to store the historical data for further analysis and reporting. Options include SQL databases (e.g., PostgreSQL, MySQL) or NoSQL databases (e.g., MongoDB, Cassandra).

- Implement data retention policies to manage and archive historical data.

8. Data Visualization:

- Create a front-end web application or mobile app to display real-time and historical temperature and humidity data in an intuitive way.

- Use visualization libraries and frameworks like D3.js, Chart.js, or Plotly for interactive charts and graphs.

9. Alerts and Notifications:

- Implement an alerting system that triggers notifications when predefined thresholds are exceeded. Use services like Amazon SNS or Twilio for SMS, email, or push notifications.

10. Scalability:

- Ensure that the platform is scalable to accommodate a growing number of IoT devices and users. Use cloud services that provide auto-scaling capabilities.

11. Monitoring and Maintenance:

- Implement monitoring and logging tools (e.g., Prometheus, Grafana, ELK stack) to track the health and performance of your platform.

- Regularly update and maintain both the server and IoT device firmware to ensure security and functionality.

12. Compliance and Regulations:

- Depending on the use case and location, consider compliance with data privacy regulations such as GDPR or HIPAA.

13. Documentation:

- Create comprehensive documentation for both the IoT device setup and the platform's API for users and developers.

14. Testing and Quality Assurance:

- Perform rigorous testing, including unit tests, integration tests, and stress tests, to ensure the reliability and security of your platform.

15. User Support:

- Provide user support, including FAQs, knowledge base, and a contact point for users who encounter issues or have questions.

This comprehensive approach should help you design a platform to receive and display real-time temperature and humidity data from IoT devices for environmental monitoring, with a focus on security, scalability, and user-friendly data presentation.