Security Considerations for Weather App Deployment

1. Authentication & Authorization

- Implement OAuth 2.0 / OpenID Connect for secure authentication.
- Use role-based access control (RBAC) for managing user permissions.
- Secure API endpoints using JWT tokens or session-based authentication.

2. Data Encryption

- Encrypt sensitive user data in transit using TLS 1.2+.
- Encrypt data at rest using AES-256 for databases and storage services.
- Use HTTPS for all API communications to prevent man-in-the-middle attacks.

3. Rate Limiting & DDoS Protection

- Implement API rate limiting at the gateway level to prevent abuse.
- Use Web Application Firewalls (WAF) to filter malicious traffic.
- Deploy bot mitigation techniques to prevent automated attacks.

4. Input Validation & Sanitization

- Validate all user inputs to prevent SQL injection, XSS, and CSRF attacks.
- Sanitize data before storing or processing it.
- Use prepared statements and ORM libraries for database queries.

5. Secure Dependencies & Packages

- Regularly update dependencies to patch known vulnerabilities.
- Use package managers with vulnerability scanning (e.g., `npm audit`, `pip audit`).
- Restrict unnecessary third-party libraries to reduce attack surface.

6. Secure API Gateway

- Enforce authentication at the API Gateway level.
- Implement request validation to block malformed or unexpected requests.
- Log API request details for anomaly detection.

7. Database Security

- Use separate SQL and NoSQL databases with the principle of least privilege.
- Disable direct database access from the public internet.
- Enable database auditing and logging for security monitoring.

8. Secrets Management

- Store API keys, database credentials, and tokens securely using environment variables or secret management tools like AWS Secrets Manager or HashiCorp Vault.
- Rotate secrets regularly to minimize risks.

9. Logging & Monitoring

- Use centralized logging (ELK Stack, Prometheus, Grafana) to monitor security events.
- Detect and alert on suspicious activities in real-time.
- Implement audit trails to track user and system actions.

10. Container & Kubernetes Security

- Use minimal base images for Docker containers to reduce vulnerabilities.
- Scan container images for vulnerabilities before deployment.
- Enforce network policies in Kubernetes to isolate microservices.
- Enable Role-Based Access Control (RBAC) in Kubernetes.

11. Cloud & Infrastructure Security

- Implement IAM policies with the principle of least privilege.
- Enable logging and monitoring for cloud resources.
- Use security groups and firewalls to restrict access.

12. Incident Response Plan

- Set up automated alerts for security breaches.
- Have a response plan to mitigate threats and recover services.
- Conduct periodic security audits and penetration testing.