Security Considerations for Smart Saver Application

### As with any application, need to consider the security of the application to ensure the customer data (PII) within it is secure both in transit and at rest. The application itself will need both front and back-end security controls plus the original source code must be secure and coded to latest security standards.

1. **Network Security**:
   * **Virtual Network Integration**: Deploy the app into a private virtual network (VNet) to isolate it from public internet traffic.
   * **Network Security Groups (NSGs)**: Use NSGs to restrict and monitor traffic by port, protocol, source IP address, or destination IP address.
2. **Identity and Access Management (IAM)**:
   * **Role-Based Access Control (RBAC)**: Implement RBAC to ensure that only authorized users have access to the app and its resources.
   * **Azure Active Directory (AAD)**: Use AAD for user authentication and authorization.
3. **Data Security**:
   * **Encryption**: Encrypt data at rest and in transit to protect sensitive information.
   * **Azure Key Vault**: Store and manage cryptographic keys, secrets, and certificates securely.
4. **Container Security**:
   * **Image Security**: Use a private container registry and scan container images for vulnerabilities before deployment.
   * **Runtime Security**: Monitor and secure running containers to detect and respond to threats.
5. **Compliance and Monitoring**:
   * **Compliance**: Ensure the app complies with relevant regulations and standards, such as GDPR and PCI DSS.
   * **Monitoring and Logging**: Use Azure Monitor and Log Analytics to track security-related activities and detect anomalies.
6. **Application Security**:
   * **Secure Coding Practices**: Follow secure coding practices to prevent common vulnerabilities such as SQL injection and cross-site scripting (XSS).
   * **Regular Updates**: Keep the application and its dependencies up to date with the latest security patches.
7. **Logging and Monitoring**:
   * **Centralized Logging**: Implement centralized logging for all components (Docker, Flask, React) to monitor and detect suspicious activities.
   * **Monitoring Tools**: Use monitoring tools like Prometheus and Grafana to keep an eye on the health and performance of your application.
8. **CI/CD Security**:
   * **Secure Pipelines**: Ensure your CI/CD pipelines are secure and only trusted code is deployed.
   * **Automated Testing**: Implement automated security testing (e.g., static code analysis, dependency scanning) in your CI/CD pipelines.