SI Docflow Admin

1. **Target Environment**
    - Linux VM in Cloud
    - Hardware requirements (CPU, RAM, storage, GPU, etc.)
        i. Min 2 CPU cores (scaled with usage)
        ii. Min 1GB RAM (scaled with usage)
        iii. Min 1GB storage (scaled with usage)
        iv. No GPU is needed
2. **Software Dependencies**
    - Node.js v18+
    - React (create-react-app)
    - Containerization handled by cloud provider
3. **Installation and Configuration**
    - No installation needed, JS code is interpreted on server as needed
    - Environment variables (*.env*):
        i. DATABASE_URL
        ii. SESSION_SECRET
    - Steps to initialize the system:
        i. Clone latest version from dev branch
        ii. npm run dependencies && CI= npm run build
        iii. npm run start
    - No default users or passwords are provided
4. **Continuous Integration / Continuous Deployment (CI/CD)**
    - No CI/CD tools are used on code repository, latest version is pulled by cloud provider
    - After pushing to *dev* branch, latest version must be approved on cloud provider

5. **Network and Security**

    - Default Port 5000 (can be different based on cloud provider)
    - SSL/TLS requirements are handled by cloud provider automatically (HTTPS with TLS certificates)

    - Authentication mechanisms in app:
        i. OAuth 2.0 with 3rd party providers
        ii. JWT with fixed duration

6. **Database Deployment**

    - PostgreSQL is required DBMS

    - Initialization scripts can be run with *npm run migrate* from *si-docflow-admin/backend*

    - Hosting can be local, cloud or a managed DB
      Depends only on DATABASE_URL connection string in *.env*

7. **Rollback and Recovery**

    - Deployment rollback depends on cloud provider

    - Backup procedures not needed as there is no production data of meaningful value

8. **Monitoring and Logging**

    - Runtime logs are generated and printed to deployment console while data transactions are logged and stored in appropriate DB table

9. **User Access and Roles**

    - Who can access the deployed system?
        i. Users who pass login in deployed application
        ii. Development team members with production DB connection access

    - How are users provisioned and managed?
        i. Handled in DB
        ii. Currently any user can register as initial development is in progress

10. **Testing in Deployment Environment**

- Smoke testing or sanity check procedures post-deployment are currently not automatic

11. **Step-by-step deployment to blank environment**

- Step-by-step guide on environment and system setup:
    - Pull project from si-docflow-admin
    - Create .env file with DATABASE_URL and SESSION_SECRET
    - Set DATABASE_URL to a valid PostgreSQL DB (local or remote)
    - Set SESSION_SECRET
    - Run command: *npm run dependencies*
    - Run command: *npm run build*
    - From *./backend* run command: *npm run migrate* (if DB is not set up already)
    - Run command: *npm run start*

SI Docflow Server

12. **Target Environment**

    - Operating System(s):
        i. Linux-based environment.

    - Hardware requirements:
        i. 1 GB RAM / 2 vCPU per service

    - Cloud vs. on-premise:
        i. cloud

13. **Software Dependencies**

    - Required runtimes:
        i. Node.js 18+

    - Libraries, packages, or frameworks:
        i. Tesseract.js
        ii. Google Vision
        iii. OpenAI SDK (for ChatGPT)
        iv. Express.js
        v. Sequelize ORM

    - Containerization: Not containerized.

14. **Installation and Configuration**

    - How is the software installed?

      No installation needed, JS code is interpreted on the server as needed.

    - Environment variables or config files required?
        i. PORT
        ii. DATABASE_URL
        iii. GOOGLE_CREDENTIALS_BASE64
        iv. GOOGLE_APPLICATION_CREDENTIALS
        v. OPENAI_API_KEY
        vi. AI_MODEL_NAME
        vii. AI_MODEL_DOWNLOAD_URL
        viii. SUPABASE_URL

ix. SUPABASE_SERVICE_ROLE_API_KEY

- Steps to initialize the system?
  i. clone from dev branch
  ii. set .env
  iii. npm install
  iv. npm run build
  v. npm run start

- Default users and passwords?

  No default users or passwords are provided.

15. **Continuous Integration / Continuous Deployment (CI/CD)**

- Are CI/CD tools used?
  i. No.

- What triggers deployment?
  i. Commit to dev branch.

16. **Network and Security**

SSL/TLS requirements are handled by cloud providers automatically (HTTPS with TLS certificates).

17. **Database Deployment**

- DBMS required:
  i. PostgreSQL

- Initialization scripts or migrations:
  i. Handled via si-docflow-admin.

- Hosting supported:
  i. Handled via si-docflow-admin.

18. **Rollback and Recovery**

- How can the deployment be rolled back?

Depends on the cloud provider.

- Backup procedures?

  Backup procedures are not needed as there is no production data of meaningful value.

19. **Monitoring and Logging**

Runtime logs are generated and printed to the deployment console while data transactions are logged and stored in the appropriate DB table.

20. **User Access and Roles**

- Who can access the deployed system?
  i. Development team members with production DB connection access.

- How are users provisioned and managed?
  i. Handled in DB.

21. **Testing in Deployment Environment**

Smoke testing or sanity check procedures post-deployment are currently not automatic.

22. **Step-by-step deployment to blank environment**

- clone from si-docflow-server
- create .env file and set variables
- run command npm install
- run command npm run build
- run command npm run start

SI Docflow Windows

23. **Target Environment**

    - Operating system(s):
        i. Windows 10 or newer

    - Hardware requirements:
        i. CPU: x64, 2-core
        ii. RAM: 4 GB
        iii. Storage: 1 GB

    - Cloud vs. on-premise:
        i. on-premise (installed and runs locally, communicates with remote APIs)

24. **Software Dependencies**

    - Required runtimes:
        i. .NET 6 or later

    - Libraries, packages, or frameworks:
        i. WinUI 3
        ii. Windows App SDKt

    - Containerization: Not containerized.

25. **Installation and Configuration**

    - How is the software installed?

      The app is launched by running the executable from the Release folder.

    - Environment variables or config files required

      The app depends on configuration files located alongside the executable.

    - Steps to initialize the system?
        i. Copy the entire Release folder to the target machine.
        ii. Run the .exe to start the app.

    - Default users and passwords?

No default users or passwords.

26. **Continuous Integration / Continuous Deployment (CI/CD)**

27. **Network and Security**

    - Ports that need to be open on each system:

      The application requires the port used for API communication to be open. This port is forwarded on the router and must be allowed through the Windows Firewall.

    - SSL/TLS requirements:

      SSL/TLS is enforced on the API server side (hosted externally). The application communicates securely via HTTPS.

    - Authentication mechanisms (e.g., OAuth, JWT):

      Authentication is handled by the external API servers, not by the application itself.

28. **Database Deployment**

29. **Rollback and Recovery**

30. **Monitoring and Logging**

    - Tools used:

      No dedicated monitoring tools.

    - Logs generated and their location:

      Logs are available in the debug console during runtime.

      Additionally, logs are sent via APIs to external servers for storage and analysis.

31. **User Access and Roles**

32. **Testing in Deployment Environment**

33. **Step-by-step deployment to blank environment**

Answer the questions as if deployment environment is non-existing (from scratch)

34. **Target Environment**
    - Operating System(s) (e.g., Windows, Linux, macOS)
    - Hardware requirements (CPU, RAM, storage, GPU, etc.)
    - Cloud vs. on-premise (e.g., AWS, Azure, Local server)
35. **Software Dependencies**
    - Required runtimes (e.g., Java JDK, .NET, Node.js)
    - Libraries, packages, or frameworks (e.g., TensorFlow, Flask, React)
    - Containerization (e.g., Docker image and version)
36. **Installation and Configuration**
    - How is the software installed? (e.g., executable, script, Docker container)
    - Environment variables or config files required
    - Steps to initialize the system (e.g., database setup, API keys)
    - Default users and passwords
37. **Continuous Integration / Continuous Deployment (CI/CD)**
    - Are CI/CD tools used? (e.g., GitHub Actions, Jenkins)
    - What triggers deployment? (e.g., commit to `main` branch)
38. **Network and Security**
    - Ports that need to be open on each system
    - SSL/TLS requirements
    - Authentication mechanisms (e.g., OAuth, JWT)
39. **Database Deployment**

- DBMS required (e.g., PostgreSQL, MongoDB)
- Initialization scripts or migrations
- Hosting supported (local, cloud, managed DB)

40. **Rollback and Recovery**
    - How can the deployment be rolled back?
    - Backup procedures (especially for databases)

41. **Monitoring and Logging**
    - Tools used (e.g., Prometheus, ELK Stack)
    - Logs generated and their location

42. **User Access and Roles**
    - Who can access the deployed system?
    - How are users provisioned and managed?

43. **Testing in Deployment Environment**

- Smoke testing or sanity check procedures post-deployment

44. **Step-by-step deployment to blank environment**

- Prepare step-by-step guide on environment and system setup (19.05.2025)
- Prepare a video for 26.05.2025.