Schedool

Software Deployment Plan

[Latest update: 2025-10-21]

Team : Win, Beam, Poon, May

1. System Requirements

Supported Environments

- **Windows:** Windows 10 and above
- **macOS:** macOS Monterey and above (latest two versions)
- **Linux:** Ubuntu 18.04+, openSUSE Leap 15+

Hardware Requirements

- Minimum 4 GB RAM
- At least 200 MB of available disk space
- 1 GHz dual-core processor or better
- Local disk access for database file storage

Software Requirements

- Node.js version 18 or higher
- npm or yarn package manager
- SQLite 3 (included or installed via package manager)

2. Deployment Strategy Summary

ScheDool will be distributed as a compressed ZIP package containing all project files. Users will extract the ZIP file, install the required dependencies using npm, and launch the app via a local GUI.

The app runs completely offline, using SQLite as a local database stored in the same directory as the application files.

Key Tools Used:

- **React.js** for frontend UI
- **TypeScript** for backend logic
- **Prisma ORM** for database management
- **SQLite** for persistent data storage
- **npm** for dependency installation and script execution

Form of Distribution Kit:

- ScheDool.zip
    - Source code (frontend + backend)
    - SQLite database (`scheDool.db`)
    - Installation instructions (`README.txt`)
    - User manualDemo data

3. Installation Package Contents

3.1 Required source or compiled files

- `/src` – frontend (React, HTML, CSS, JS)
- `/server` – backend (TypeScript compiled to JS)
- `/database/scheDool.db` – local SQLite database
- `/index.html` – main application entry file

3.2 Required third-party components

- React.js
- Prisma ORM
- SQLite
- Node.js modules (as listed in `package.json`)

3.4 Required graphical assets, configuration and other non-program files

- `/assets/` – icons and UI images
- `/config/` – configuration files for database or environment
- `/data/demo_rooms.sql` – demo data for testing

3.5 Documentation files to be provided

- `UserManual.pdf` – step-by-step user guide
- `README.txt` – installation and quick start instructions
- `LICENSE.txt` – license and usage permissions

3.6 Development files and components that must be excluded

- `/tests/` – unit test files
- `/node_modules/` – can be regenerated

- `.env`, `.gitignore`, `.vscode/`
- Temporary logs or build artifacts

4. Additional Code Required for Deployment

The following scripts are used to assist in deployment and packaging:

- `build.sh` / `build.bat` – automates packaging into a ZIP file
- `init_db.ts` – initializes or resets the SQLite database
- `load_demo_data.ts` – imports demo data for presentation
- `start_local_server.js` – starts the local GUI instance

5. Deployment Tasks

- 1. Verify Node.js and npm are installed | Required for all OS
- 2. Extract ScheDool.zip | Unpack source and assets
- 3. Run 'npm install' to install dependencies | Installs React, Prisma, SQLite modules
- 4. Initialize database with 'npm run init-db' | Creates SQLite tables and demo data
- 5. Run 'npm run start' | Launches local GUI
- 6. Confirm database and UI load correctly | Manual verification
- 7. Package app into new ZIP for distribution | Prepare final deliverable
- 8. Test deployment on Windows, macOS, and Linux | Ensure cross-platform compatibility
- 9. Finalize documentation (User Manual + README) | Include in distribution package

6. Deployment Test Plan

- 1. Install dependencies on Windows | No missing package errors
- 2. Launch GUI after installation | Application opens and displays weekly schedule
- 3. Create, edit, and delete reservations | Data updates correctly in SQLite
- 4. Detect conflict between overlapping reservations | System blocks duplicate booking
- 5. Export weekly schedule to file | File exported as .pdf or .txt
- 6. Run on macOS and Linux | App installs and launches successfully
- 7. Check database persistence after restart | Data remains intact after closing and reopening app
- 8. Verify demo data loads correctly | Default schedules appear on initial setup