Clingies App - Technical Specification Document

Overview

Clingies is a lightweight, cross-platform desktop application for creating and managing sticky note-like items ("clingies"). Designed for productivity and quick access, the app will start minimized in the system tray and provide an intuitive, minimal UI for managing notes. It targets Windows and Linux platforms using Avalonia UI and aims to be modular and extensible for future features such as reminders, scheduling, and note recovery.

Goals and Objectives

- Create a clean, responsive, cross-platform UI using Avalonia.
- Persist clingy data using a lightweight local database.
- Run minimized in the system tray with configurable behavior.
- Design architecture for extensibility and maintainability.
- Allow lightweight ORM-like data access via Dapper.
- Support migrations and code-first DB schema management.

Technology Stack

UI Framework:	Avalonia UI
Database:	SQLite
Data Access:	Dapper
Migrations:	FluentMigrator or DbUp
Language:	C# (.NET 8)
Architecture:	Vertical Layered
Tray Integration:	Avalonia + Native APIs
Dependency Injection:	Built-in .NET DI

Feature Set (Phase 1)

Core Features:

- Create, edit, and delete clingies.
- Persist clingies between sessions.
- Tag clingies (optional).
- Display clingies as floating, draggable windows.

- Launch minimized to system tray.
- Multi-monitor support.

Stretch Goals:

- Clingy reminders.
- Programmed clingies.
- Trash/recycle bin.
- Export/import.
- Theme customization.
- Global shortcut.
- Cloud backup.

Architecture

The application will follow a vertical slice architecture where features are encapsulated as end-to-end units.

Layered Responsibilities:

- UI Layer: Avalonia views and view models.
- Application Layer: Use cases and orchestrators.
- Domain Layer: Core models and business logic.
- Infrastructure Layer: Persistence, tray integration, migration engine.

Data Model (Initial)

```
public class Clingy
{
    public Guid Id { get; set; }
    public string Content { get; set; }
    public string? Tag { get; set; }
    public DateTime CreatedAt { get; set; }
    public DateTime? ModifiedAt { get; set; }
    public bool IsDeleted { get; set; }
}
```

Clingies App - Technical Specification Document

Database & Migrations

Database: Local SQLite file (e.g., clingies.db).

Migrations: Use FluentMigrator or DbUp.

- Code-first schema changes.
- Schema version tracking.
- Runs on app startup or dedicated tool.

Advantages:

- No need for full EF Core.
- Simple integration with Dapper.

Persistence Strategy

Dapper Repositories with parameterized SQL.

Encapsulate SQL within feature slices.

Example:

```
public async Task<Clingy?> GetByIdAsync(Guid id)
{
    using var conn = new SQLiteConnection(_connectionString);
    return await conn.QuerySingleOrDefaultAsync<Clingy>(
        "SELECT * FROM Clingies WHERE Id = @Id AND IsDeleted = 0", new { Id = id });
}
```

System Tray Integration

Use Avalonia AppBuilder for minimized start.

System tray via Avalonia Community Toolkit or native interop.

Tray Behavior:

- Launch to tray.
- Double-click opens clingy list or new clingy.

- Right-click menu: New, Settings, Exit.

Testing

Unit Testing: Core features and logic.

UI Testing: Avalonia testing framework.

Integration Testing: Data access and migrations.

Build and Deployment

Build System: .NET CLI + MSBuild.

Packaging:

- Windows: MSIX or EXE.
- Linux: .deb and AppImage.

CI/CD: GitHub Actions for multiplatform builds.

Suggestions

Settings System: JSON config.

Telemetry: Opt-in usage tracking.

Plugin System: For extensibility.

Rich Text Support: Markdown for formatting.

Conclusion

Clingies aims to be a lightweight, robust, and visually intuitive tool to manage personal notes across platforms. The chosen stack offers maximum flexibility and minimal overhead, paving the way for rapid feature iteration and a pleasant user experience.