Developer Manual for Student Tracker

Version: 1.0

Last Updated: 12/01/2025

1. Glossary

- Blazor: The .NET web framework used for building the client-side UI.
- DbContext: The Entity Framework Core session with the database.

2. Development Environment & Setup

Prerequisites

- Framework: .NET 8.0
- IDE: Visual Studio 2022 Community Edition
- Database: SQLite

Setup Instructions

1. “git clone https://github.com/SE1-Group-8/StudentTracker” or your preferred method of cloning a repo.
2. Open StudentTracker.sln in Visual Studio.
3. Restore Packages.
4. Run the Application, which you can hit “F5” to do.

3. Coding Standards

- Naming Conventions:
    - Classes/Methods: PascalCase
    - Variables/Parameters: camelCase
    - Interfaces: Prefix with ‘I’
- Asynchronous Programming: All I/O bound database operations must use async/await.
- Files: One class per file.

4. Development Standards

- Architecture: Separation of concerns using the Repository Pattern. Controllers/UI should never access ApplicationDbContext directly, they should go through one of the preexisting methods.
- State Management: User state is handled via UserSession.cs.
- Version Control:
    - Main branch is where most of the changes are made.
    - Release branch
- Security: Passwords must be hidden when typed.

5. Architectural Design Document

- Link to the Architectural design document: Architectural Design Document

6. Detailed Design Document

- Link to the Detailed design document: Detailed Design Document

7. Test Process

- The unit tests and build tests are automatically done through github actions.

8. Issue Tracking Tool

- Tool: GitHub Issues
- Reporting Bugs: Try your best to include steps to reproduce and what browser version you used.

9. Project Management Tool

- Trello
- Sprint Cycle: 1 week for 5 weeks

10. Build and Deployment (Procedure)
    - Github provides the source code and github actions makes sure that it is successfully buildable on the major operating systems.
11. Rationale Behind Design Decisions
    - Repository Pattern:
        - Decision: We used repositories to separate the UI from the Database layer. This allowed for easier unit testing.
    - Blazor:
        - Decision: There was some debate on what framework we wanted to use but ultimately everyone was at least somewhat familiar with Blazor and no other framework, so Blazor is what we went with.
    - SQLite:
        - Decision: Chosen initially to minimize the learning time of a new database and to get straight into working on the project.
12. Troubleshooting Guide
    - Currently there are no known issues with the application. If you encounter any problems, please report them by adding an entry to the Issues section on the GitHub.