**B. Conception & Organization**

**B1. Software Conception (2 pts)**

The Extreme Minimalism AI Coach follows a layered architecture that keeps sensitive coaching data local while providing adaptive guidance:

* **Client experience:** Three single-page views (assessment, dashboard, chat) bootstrap through a shared authentication overlay. The overlay negotiates login, unlocks the encrypted vault, and emits vault:updated events so each view reacts to fresh progress, goals, and conversation history.
* **Server services:** A Node.js/Express application handles routing, authentication, and vault lifecycle management. Express-session and JWT middleware guard routes; vault helpers serialize and encrypt user data on every write. The AI orchestration layer assembles prompts, detects emotional cues, and streams responses from the GPT4All runtime.
* **Data model:** User credentials (id, email, password hash, role, salts) are stored separately from coaching content. Each vault file contains the profile, progress milestones, attachment stories, and conversation history, all wrapped in AES-256-GCM using keys derived from the user’s password. Progress records track phases, target counts, milestones, and timestamps, enabling analytics without exposing personal text.
* **Integration flow:** When the chat UI submits a message, the server determines the appropriate coaching approach, enriches the prompt with vault context and emotional state, invokes GPT4All locally, and streams tokens back to the client. Sequence diagrams confirm that vault synchronization and AI responses remain consistent across sessions.

**B2. Project Organization Tooling (1 pt)**

Work was planned and tracked on a GitHub Project board that mirrored the course phases. Columns represented backlog, in progress, review, and done states; cards linked to implementation issues for privacy, innovation, and organization phases. Each milestone corresponded to a tagged release (v0.5.0 through v1.0.0), providing checkpoints that matched classroom deliverables. This structure made it easy to visualize pending work, attach documentation drafts, and record demo preparations.

**B3. Feedback on Self-Organization (1 pt)**

Reflecting on the workflow revealed several lessons:

* Sequencing major efforts—security first, then privacy safeguards, followed by innovation research—reduced rework because each phase built on the protections established earlier.
* Batching the front-end refactor (introducing the auth overlay and shared vault synchronization) before layering privacy features avoided duplicate UI changes and sped up testing.
* Maintaining contemporaneous notes after each phase ensured documentation mirrored the finished product rather than aspirational plans.

Future improvements include migrating JSON storage to a managed database with migrations, expanding automated tests around vault lifecycle and privacy endpoints, and adding observability (structured logs and metrics) to monitor AI response quality over longer pilots.