# Salatiso Ecosystem: Technical Specification & Architectural Blueprint  
\*\*Document ID:\*\* ARCH-SPEC-V2.0   
\*\*Date:\*\* 2025-07-19   
\*\*Status:\*\* Adopted  
  
---  
  
### \*\*Part 1: Architectural Audit & Feedback\*\*  
  
This audit confirms that the architecture implemented in the `hub.sazi.life.v2` repository is exceptionally well-conceived and serves as a robust blueprint for all future projects.  
  
\* \*\*Overall Structure:\*\* The separation of concerns is excellent. Placing public-facing pages and the primary authenticated dashboard (`index-modules.html`) in the root, core application views in `/modules`, and reusable UI snippets in `/components` is a clean, logical, and maintainable pattern.  
\* \*\*Component-Based Approach:\*\* Using discrete files for `header.html`, `sidebar.html`, `footer.html`, etc., is the correct modern approach. It ensures that a change to one component does not unintentionally break another.  
\* \*\*Asset & Script Separation:\*\* The granular separation within `/assets/css` and `/assets/js` is highly effective. Creating subfolders for each module's logic (e.g., `/assets/js/finhelp/`) is the key to the functional isolation you require. This prevents code collisions and makes debugging exponentially easier.  
\* \*\*JavaScript Engine Concept:\*\* The idea of a global `main.js` for component loading on public pages and a `modules.js` to act as the central engine for the authenticated section is a powerful pattern. `main.js` handles the "shell" of the public site, while `modules.js` handles the "brain" of the application logic once a user is signed in.  
  
\*\*Conclusion:\*\* The architecture is exemplary. It provides the stability and predictability needed for this project and the entire ecosystem. We will proceed with this model.  
  
---  
  
### \*\*Part 2: Refined Master File Repository Structure\*\*  
  
This is the formalised blueprint based on your new repository. This structure will be the standard for all ecosystem websites.

/hub.sazi.life.v2/

|

├── index.html // Public landing & login page.

├── index-modules.html // Authenticated main dashboard view after login.

├── about.html // Generic public page.

├── contact.html // Generic public page.

|

├── /modules/ // Authenticated user-facing application sub-pages.

│ ├── profile.html // User profile page.

│ │

│ ├── /finhelp/ // FinHelp module pages.

│ │ ├── index.html

│ │ └── ...

│ │

│ └── /familyhub/ // FamilyHub module pages.

│ ├── index.html

│ └── ...

|

├── /components/ // Reusable HTML partials.

│ ├── header.html

│ ├── footer.html

│ ├── sidebar.html

│ ├── theme-switcher.html

│ └── language-switcher.html

|

├── /assets/

│ ├── /css/ // Global and component-specific stylesheets.

│ │ ├── style.css // Main site-wide styles.

│ │ └── ...

│ │

│ ├── /js/ // JavaScript files.

│ │ ├── main.js // Public Engine: Loads components on public pages (e.g., index.html).

│ │ ├── modules.js // Authenticated Engine: Manages modules after login (e.g., index-modules.html).

│ │ ├── auth.js // Handles Firebase authentication logic.

│ │ ├── firebase-config.js // Firebase initialization.

│ │ ├── translations-engine.js // Handles language switching.

│ │ │

│ │ ├── /translations/ // Language-specific data files.

│ │ │ ├── en.js

│ │ │ └── xh.js

│ │ │

│ │ └── /finhelp/ // Module-specific logic.

│ │ ├── finance-ui.js

│ │ └── finance-db.js

│ │

│ └── /images/ // Site images and icons.

|

└── /documents/ // Project planning and specification documents.

---  
  
### \*\*Part 3: Generic Technical Specification (Ecosystem Standard)\*\*  
  
This defines the phased development standard for any website built under the Salatiso brand.  
  
#### \*\*Level 1: Basic Static Website\*\*  
\* \*\*Philosophy:\*\* A professional, fast, and reliable public-facing "digital business card."  
\* \*\*Technology Stack:\*\* HTML5, CSS3, Vanilla JavaScript (ES6), Tailwind CSS.  
\* \*\*Architecture:\*\* Reusable HTML partials in `/components/` are loaded by `assets/js/main.js` into placeholder `<div>`s.  
\* \*\*Characteristics:\*\* No backend database, no user authentication, extremely fast, simple to deploy.  
  
#### \*\*Level 2: Hybrid Static & Dynamic Website (The Hub Standard)\*\*  
\* \*\*Philosophy:\*\* An authenticated web application with a static public front-end. This is the current model for `hub.sazi.life.v2`.  
\* \*\*Technology Stack:\*\* All technologies from Level 1, plus Firebase (Firestore, Authentication, Hosting).  
\* \*\*Architecture & User Flow:\*\*  
 1. \*\*Public Entry:\*\* The user lands on `index.html`, which functions as the public-facing site and login gateway.  
 2. \*\*Authentication:\*\* The `assets/js/auth.js` script handles all sign-up, sign-in, and password reset logic on `index.html`.  
 3. \*\*Redirection:\*\* Upon successful authentication, `auth.js` redirects the user to `index-modules.html`.  
 4. \*\*Authenticated Environment:\*\* The `assets/js/modules.js` script runs on `index-modules.html` and all pages within the `/modules/` directory. It checks for an active user session (protecting the pages) and loads the necessary components (header, sidebar, etc.). It also orchestrates the specific logic for each module.  
\* \*\*Key Characteristics:\*\* Secure user login, clear separation between public and private content, scalable.  
  
#### \*\*Level 3: Full Dynamic Application (Future Goal)\*\*  
\* \*\*Philosophy:\*\* A seamless, app-like user experience for highly complex platforms.  
\* \*\*Technology Stack:\*\* React or Vue.js, Firebase (Firestore, Auth, Cloud Functions).  
\* \*\*Architecture:\*\* A Single Page Application (SPA) where the UI is managed entirely by the JavaScript framework.  
\* \*\*Key Characteristics:\*\* Fluid UX with no page reloads, highly scalable, requires a build step.