GAINS APPLICATION SYSTEM ARCHITECTURE

1. Major Logical Components

Presentation Tier (Frontend)

- Homepage (with FAQ and onboarding)
- Login Page (user authentication interface)
- Tool Dashboard (Linear Regression, Bar Chart, Line Chart selection)
- Data Input Forms (manual entry and file upload)
- Results Display (visualizations and tables)
- Code Viewer (interactive R code display with copy/download)

Logic Tier (Backend)

- Authentication Controller
- Tool Controller
- Data Controller
- Analysis Controller

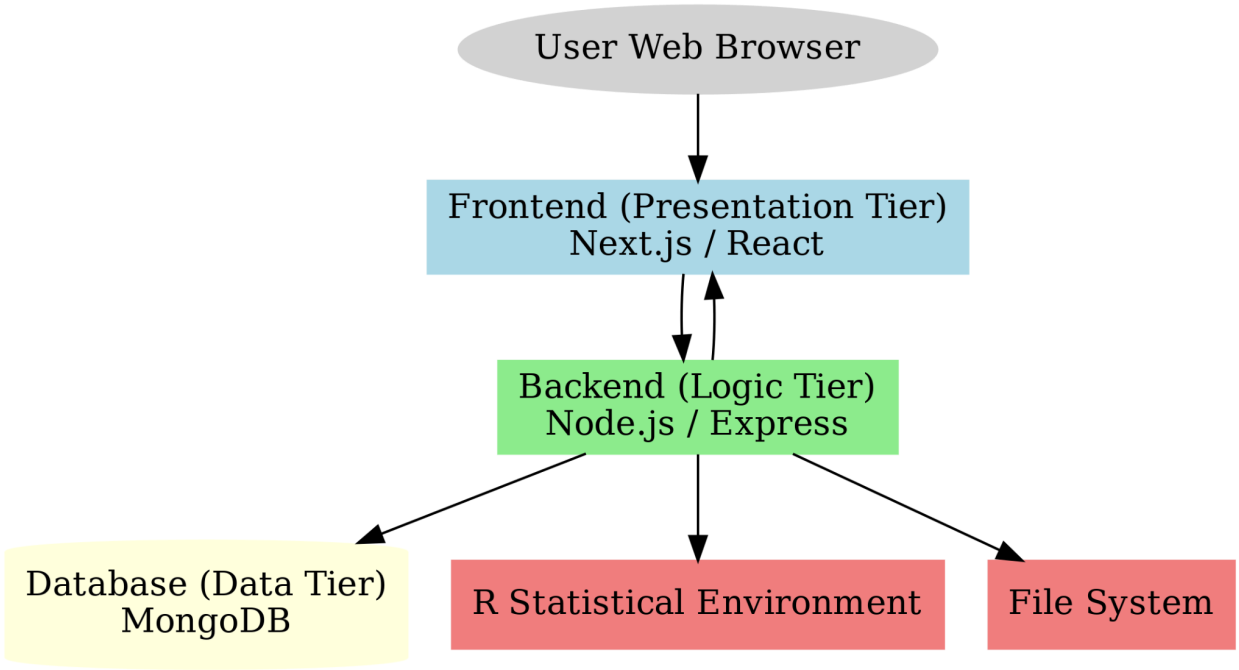
- User Service
- R Execution Service

Data Tier (Database)

- MongoDB storing:
    - User accounts
    - Statistical tool templates
    - Datasets
    - Analysis history
    - Output references

External Systems

- R Statistical Computing Environment (via Rscript)
- File System (temporary storage for datasets, scripts, and outputs)
- Web Browsers (Chrome, Safari, Firefox, Edge)

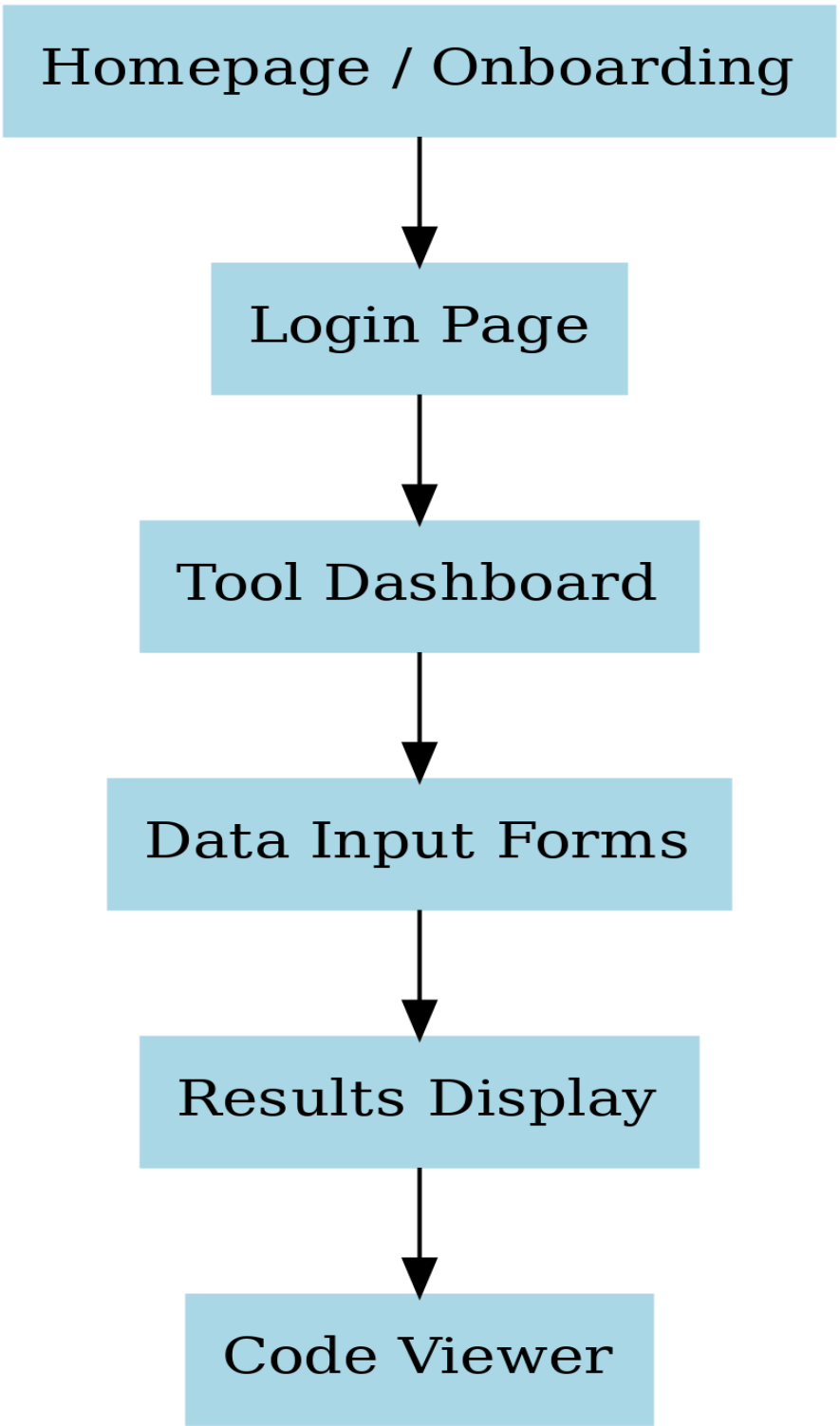


2. Role and Purpose of Each Component

Frontend Components

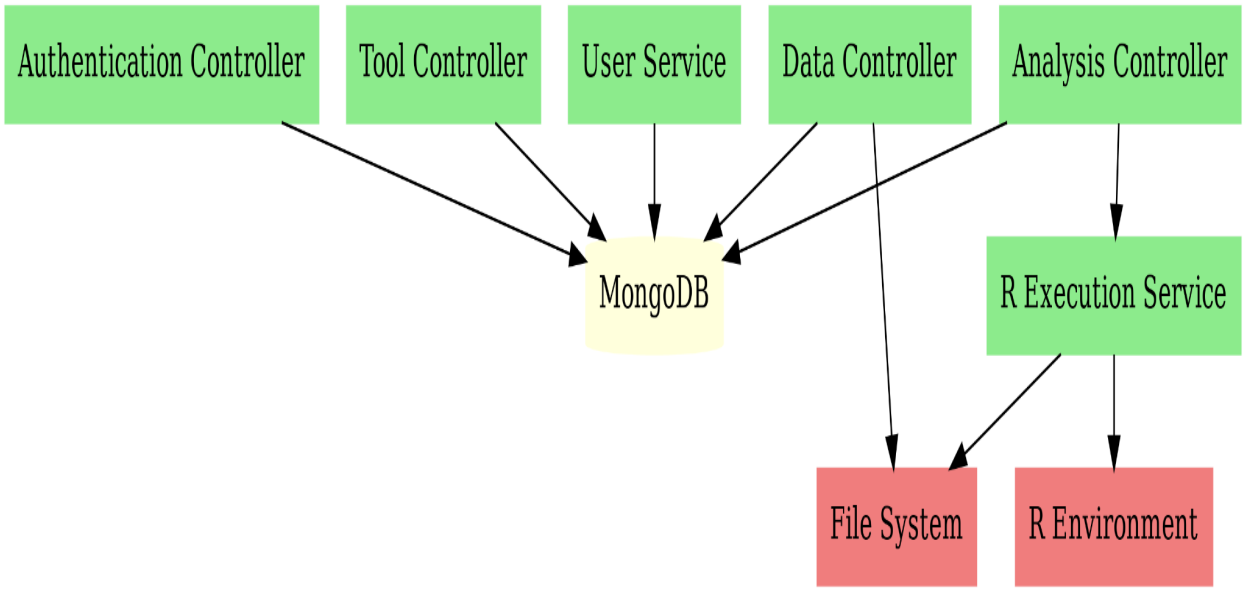
- **Homepage / Onboarding**: Provides entry point, “R Made Easy” branding, FAQs, and instructions.
- **Login Page**: Manages credential entry and session handling.
- **Tool Dashboard**: Allows users to select statistical tools, configure parameters, and view outputs.
- **Data Input**: Accepts manual inputs or uploaded datasets.
- **Results Display**: Presents visualized analysis results in charts/tables.

- **Code Viewer**: Displays generated R code snippets with options to copy or download.



Backend Components

- **Authentication Controller**: Handles user registration, login, password security, and token validation.
- **Tool Controller**: Provides predefined statistical tool configurations and metadata.
- **Data Controller**: Validates and processes uploaded datasets.
- **Analysis Controller**: Orchestrates analysis requests, executes R code, saves history, and formats results.
- **User Service**: Encapsulates business logic for managing users.
- **R Execution Service**: Interfaces with R, runs scripts securely, parses results, and handles errors.



Database

- **MongoDB**: Stores users, datasets, tool templates, and historical analyses for persistent access.

External Systems

- **R Statistical Environment**: Performs actual computation for statistical models.
- **File System**: Handles uploaded data, generated scripts, and output files.

Tools, Technologies, and Programming Languages

Frontend

- Next.js 15.5.2 (React framework with server-side rendering)
- React 19.1.0 (component-based UI library)
- JavaScript (ES6+)
- CSS Modules (scoped styling)

Backend

- Node.js (JavaScript runtime)

- Express.js (web application framework)
- JWT (authentication tokens)
- bcrypt (password hashing)

Database

- MongoDB (NoSQL database)
- Mongoose (object modeling for MongoDB)

R Execution

- Rscript (R command-line execution)
- Node child process API (to run R scripts from backend)

Development Tools

- npm (package manager)
- Git (version control)
- Visual Studio Code (IDE)

Runtime Environment

- Client: modern browsers

- Server: Node.js runtime
- Database: MongoDB instance (local or cloud-hosted)

---

4. Component Interactions and Communication

Frontend ↔ Backend (API Calls)

- Login/Signup ↔ Authentication Controller
- Tool selection/configuration ↔ Tool Controller
- Dataset upload ↔ Data Controller
- Analysis request ↔ Analysis Controller
- R code generation ↔ Analysis Controller

Backend ↔ Database (MongoDB)

- User accounts ↔ Authentication/User Service
- Tool templates ↔ Tool Controller
- Datasets ↔ Data Controller
- Analysis history ↔ Analysis Controller

Backend ↔ R Execution Service

- Analysis Controller passes generated R code and dataset →
- R Execution Service executes via Rscript →
- Results parsed and returned to backend

Backend ↔ File System

- Data Controller stores uploaded datasets
- R Execution Service manages temporary scripts and outputs

Frontend ↔ File System (Indirect)

- Users download generated R code or results through backend-served links