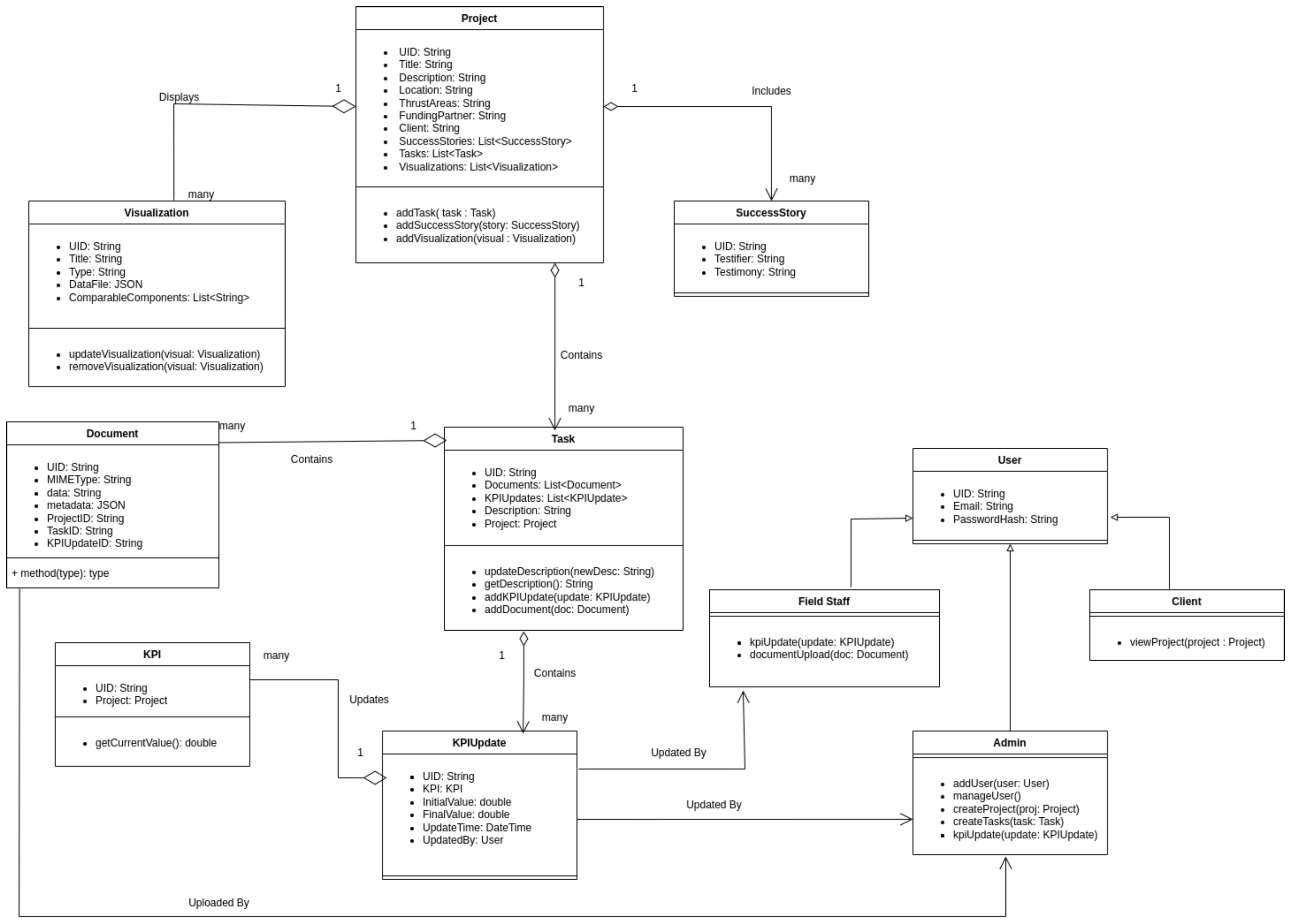
Product Design

Team 35
Gautam Bhetanabhotla, 2023101032
Saubhanik Chaudhuri, 2023115004
Nikhil Repala, 2023101084
Anvithraj Reddy Basani, 2023101023
Pallamreddy Naga Venkata Viswas Reddy, 2023101008

Design Model



Project	Class state ● UID ● Title

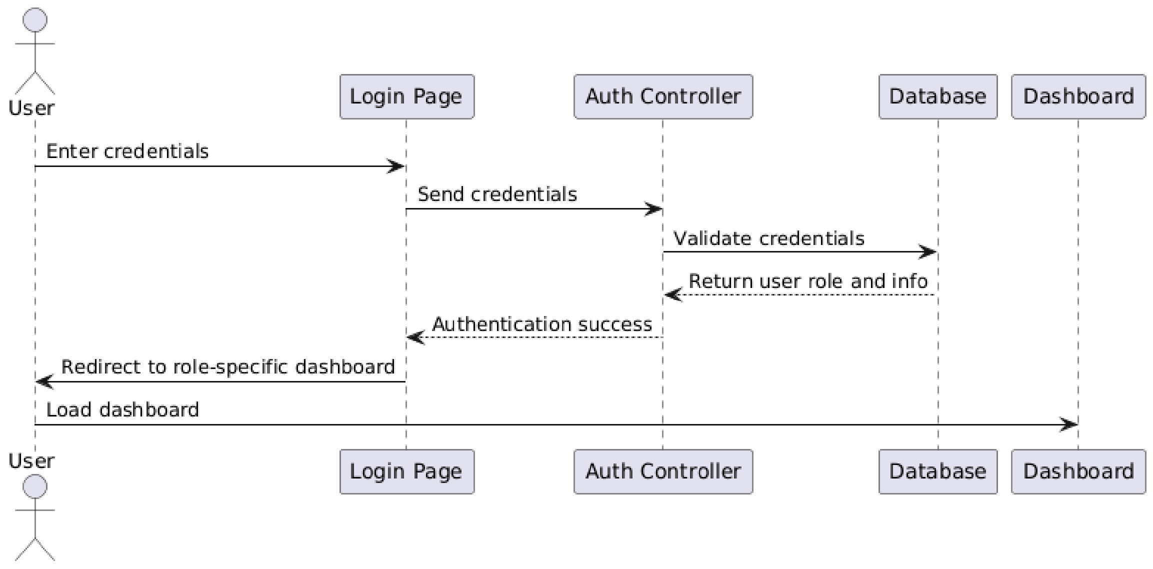
	• Description • Location • Thrust areas • Funding partner • Client • Success stories • Tasks • Visualizations Class behavior • Add task • Add success story • Add visualization
Task	Class state • UID • Array of documents • Array of KPI updates • Description • The project which it's a part of Class behavior • Update description • Get description • Add KPI Update • Add document
KPI	Class state • UID • The project which it's a part of Class behavior • Get current value
KPI Update	Class state • UID • KPI being updated • Initial value • Final value • Date and time of update • User which updated Class behavior: None
Visualization	Class state • UID • Title • Type of visualization • Data file • Comparable components Class behavior • Update Visualization • Remove visualization
User (Base Class)	Class state • UID • E-mail ID • Password Hash
Admin (Inherits User)	Class state : Inherits from Base Class User Class behaviour: • Add User • Manage User • Create Project • Create Tasks • KPI Update
Client (inherits User)	Class state : Inherits from Base Class User Class behaviour: • View Project

FieldStaff (inherits User)	Class state : Inherits from Base Class User Class behaviour: ● KPI Update ● Document Upload (Image)
Document	Class state ● UID ● Document type (jpg, pdf, etc.) ● Document content Class behavior: None
Success story	Class state ● UID ● Project it's a part of ● Testifier ● Testimony

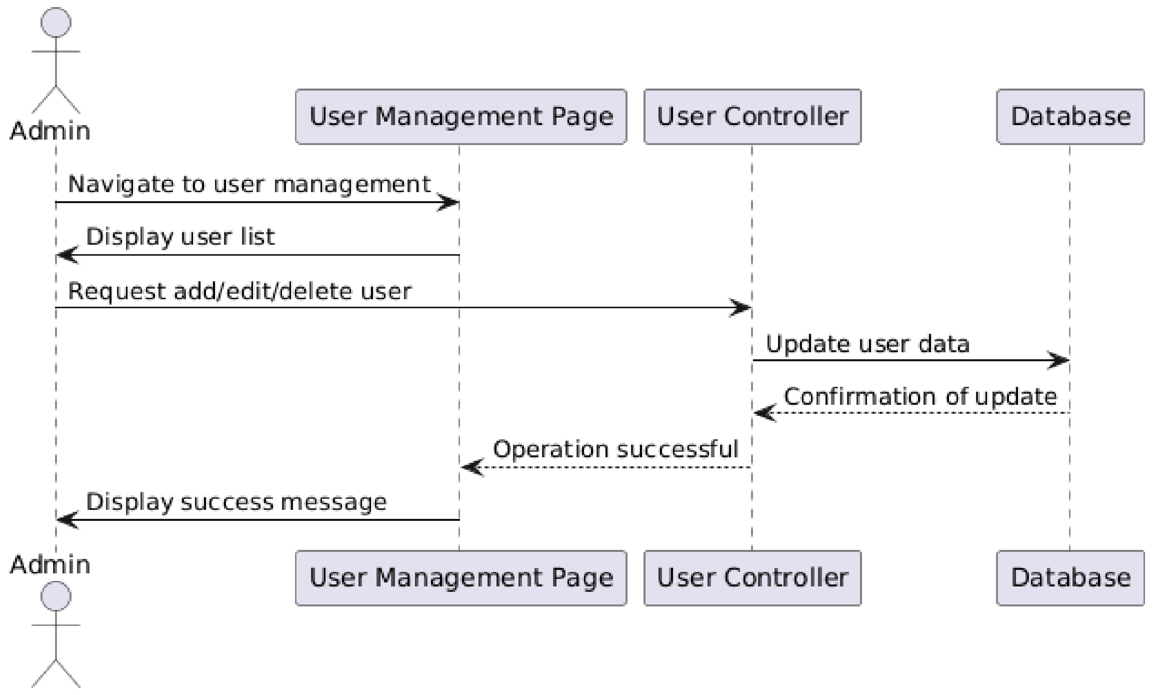
.

Sequence Diagram(s)

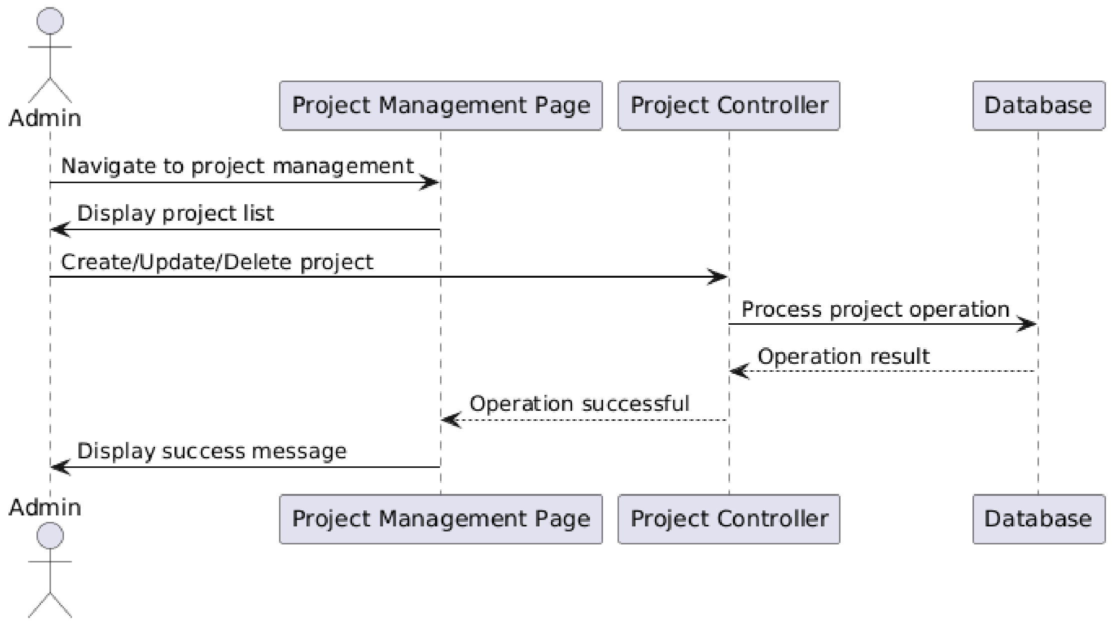
1. Role-Based Authentication



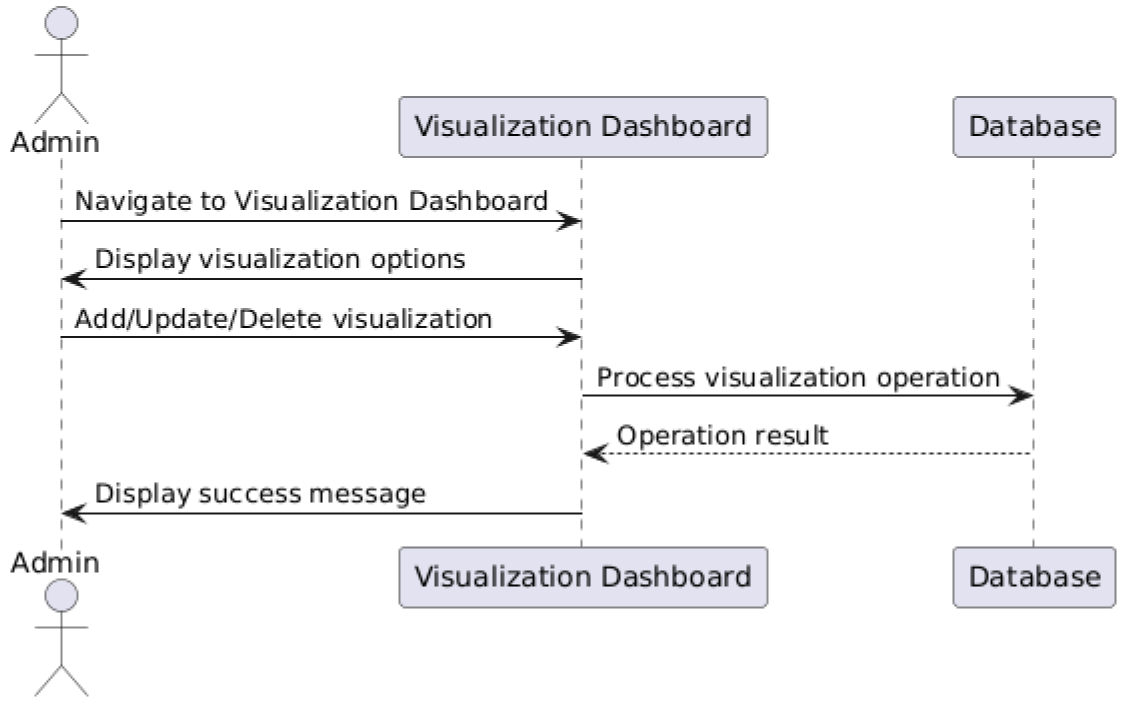
2. User Management



3. Project Management



4. Customisable Visualisations


Admin
Visualization Dashboard
Database
Navigate to Visualization Dashboard
Display visualization options
Add/Update/Delete visualization
Process visualization operation
Operation result
Display success message
Admin
Visualization Dashboard
Database

Design Rationale

Database schema

We first decided on having a mongoDB collection for each of: **User, Project, KPI, KPI Update, Task, Visualization and Document**. A project's **timeline** is a series of **Tasks** which consist of a series of **KPI updates** and uploaded **documents.** KPI updates and documents are sorted in chronological order.

Later, we decided to include KPI, KPI Update and Task as part of the project schema itself. Since they're unique to each project, there is no point storing them as separate entities and linking them through ObjectId references. However, this idea was dropped as we found deeply nested JSON objects inconvenient and also wanted to be able to individually retrieve each KPI update/Task.

Currently, we are going with the first idea again.

Update, Mar 20:
We are now storing the ID of the project that the task corresponding to a KPI update is part of. This is needed so that it's easier to write a database query to fetch all KPI updates associated with a particular project. It also makes the query more efficient.

Authentication

At first, we planned to use JWT-based authentication, but we realized that revoking access would be difficult, especially for field staff with time-limited permissions. Since JWTs remain valid until

they expire, it would be hard to remove access instantly. So, we decided to go with session-based authentication, which gives us more flexibility to revoke permissions whenever needed. This is useful, for example, when an admin is removed, as we can immediately end their session. Since the organization isn't large enough to require distributed servers, we chose security over scalability, making sessions the better option.
Also, we faced a few issues on the frontend while implementing JWTs for our Assignment 1 (the buy sell rent portal), we decided to stick to sessions as they are easier to handle.

UI Library

We initially considered DaisyUI and Material UI among many other libraries, but we liked HeroUI the most because of a wider range of components.
HeroUI gave us a bunch of problems, though, which turned out to be that HeroUI is styled with Tailwind but the latest version of HeroUI doesn't support the latest version of Tailwind. This mismatch cost us a lot of time to trace and eliminate, but we did not switch UI libraries during this time. We persisted and fixed the bug.

Chatbot

I am using Python for Chatbot to take care that no other data of other clients is used by AI for responses. I am filtering the data based on user role and providing to AI. Python is chosen for ease of use.

Document

We are storing file data as base64 strings.