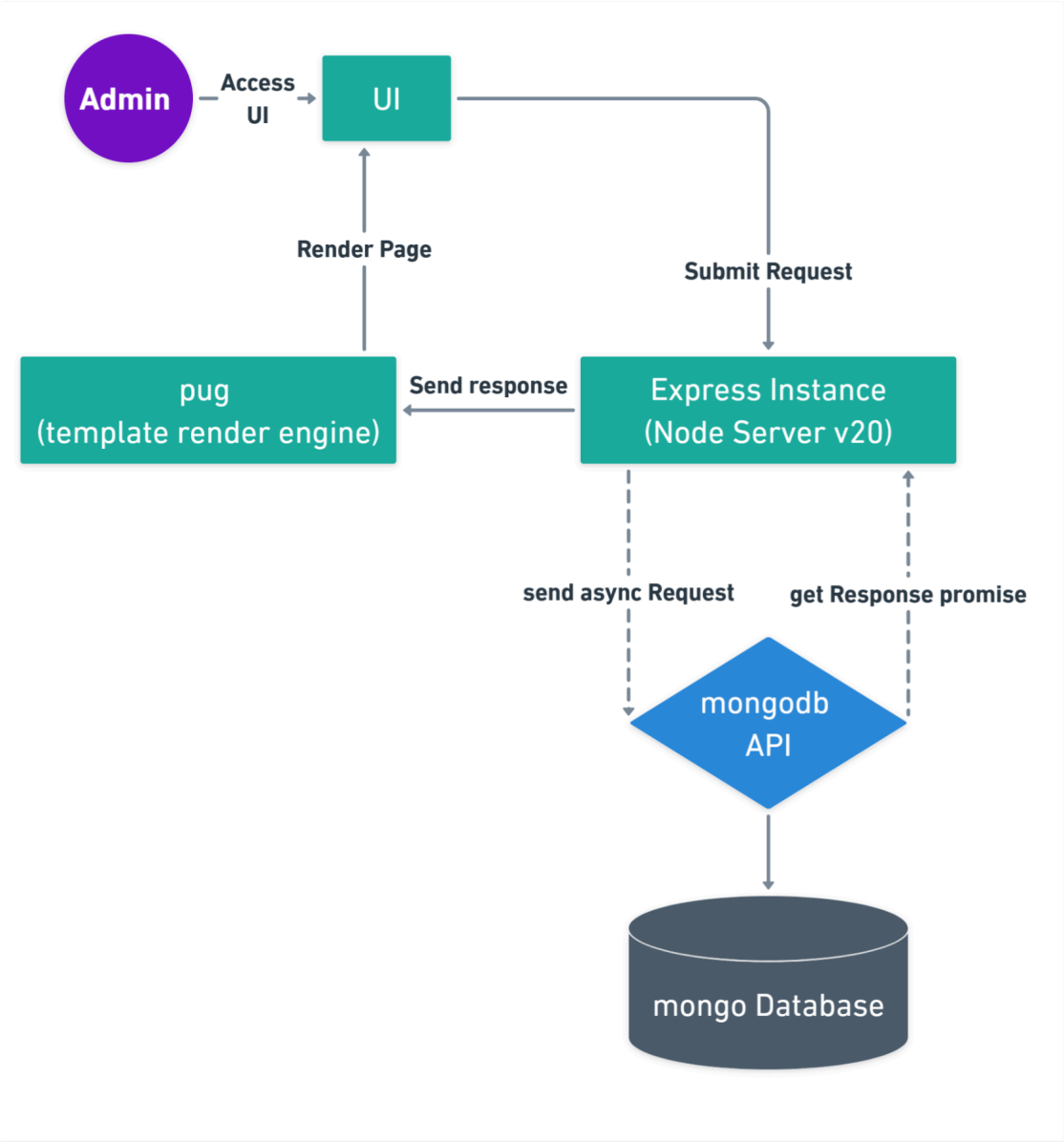
SimpleAUTH - built with MEBN(Bootstrap)

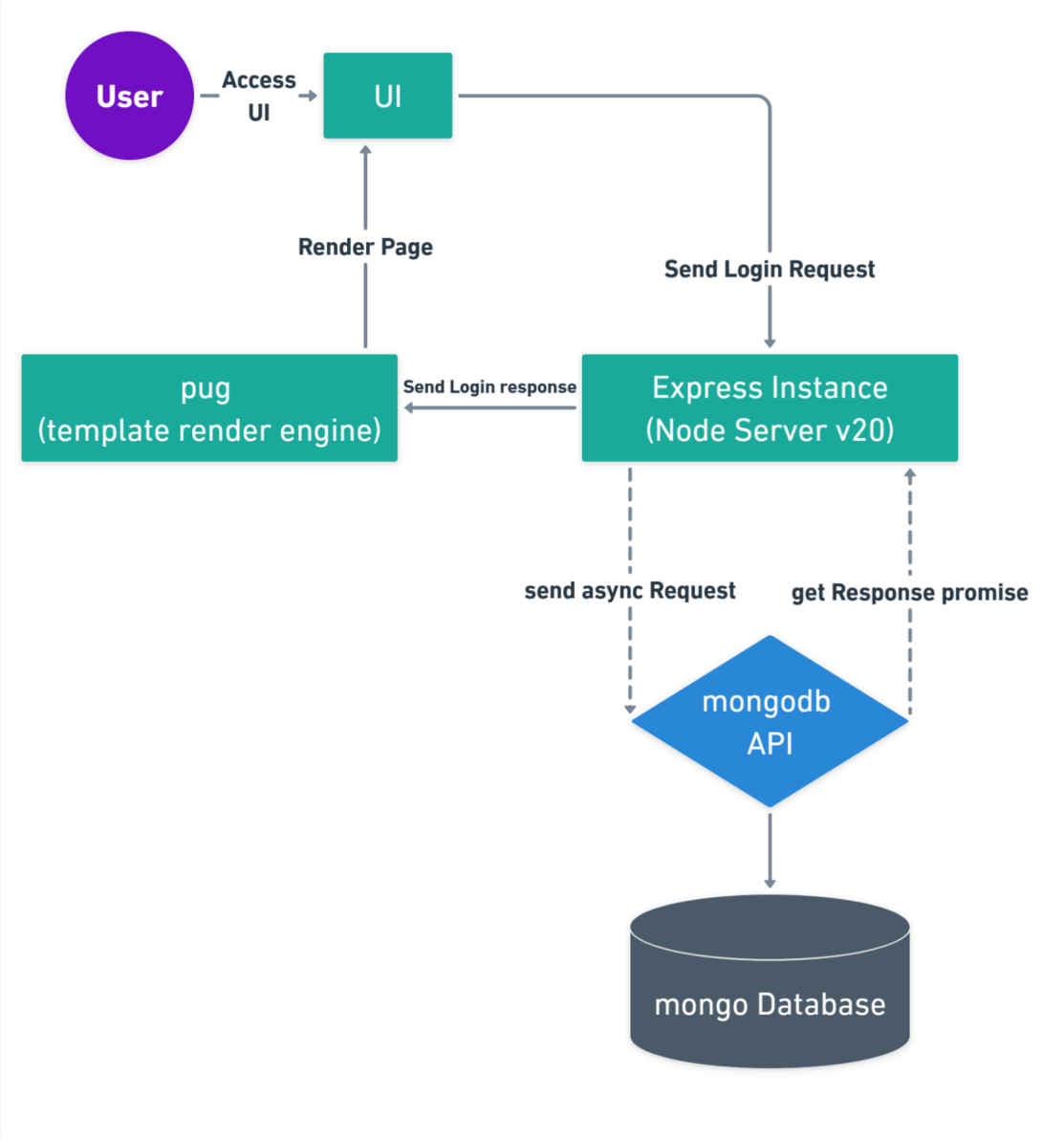
1. Backend: Outline the structure, database design, and server-side logic in a doc.

/manageUser flow diagram

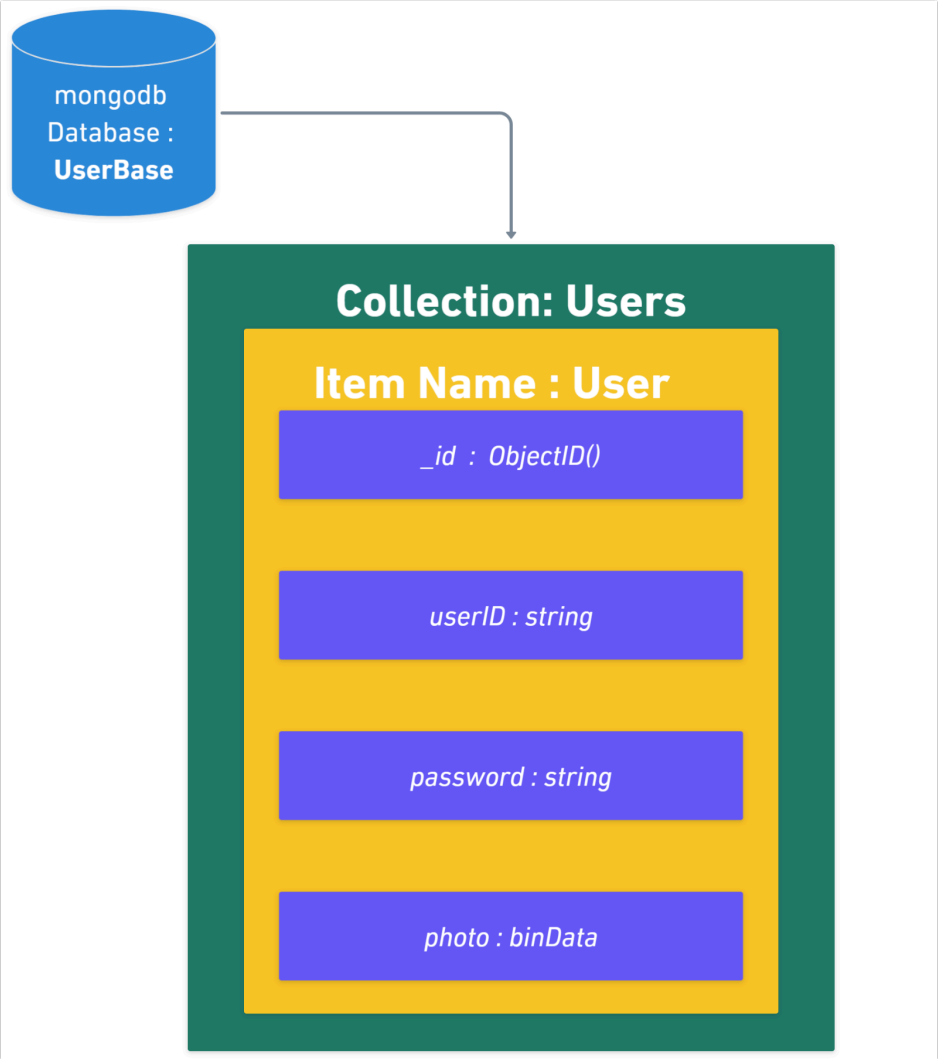


Made with Whimsical

/user - User Login Flow Diagram



MongoDB userBase Schema



Made with Whimsical

The express framework here utilized some modules to make this website:

- **Body-parser : Process Forms**
- **mongoDB: Work with MongoDB**
- **Express-session: To enable session storage for login information**

The express framework connects with a mongodb Server.

.

If the database named **“userBase”** doesn’t exist, it creates one with admin().command()

Then if the **” userBase”** exists and **“users”** collection doesnt exist, it creates one with db.createCollection() Here a validator schema was provided.

Then the collection object is globally declared for use.

The routes for the website created on express are as follows.

“/” redirects to -> /admin :

GET protocol

- The admin interacts with the UI.
- Logs in with admin credentials.
- Successfully logs in.
- Gets redirected to the createUsers page.

/processAdminForm

POST protocol

- Processes admin login credentials and redirects to “/manageUsers”

/processUserForm

POST protocol

- *Processes user login credentials and redirects to “/userHome”*

/manageUsers

get protocol

- Admin interacts with the UI
- Creates a User and sends a request to the server
- Express Server processes and sends a request to mongoDB API
- MongoDB API accesses the database and sends back a response.
- The express server processes the response and sends back a response with Pug as a rendering engine.

/user

GET protocol

- User login credentials are posted to “/processUserForm”
- Logs in with given credentials.
- Successfully logs in.
- Gets redirected to the user home page.

/userHome

GET protocol

- View User Data
- Log out and sign in as another user

2.Frontend: Discuss technologies, UI design, and user experience in a doc.

The front end uses the Bootstrap library to add styling to the webpage. It uses Pug to render dynamic content like user tables and error codes.

The code is also made to be responsive on mobiles.

The utility classes Bootstrap offered along with Pug's minimalistic html format sped up development.
Although I could've used Pug's templates and mixins to speed up development even more, I couldn't because I didn't want to overcomplicate by chance.

When designing the webpage with dynamic content, there was no easy way to display JSON output neatly without using something like Pug.

The main reason to make this website was to allow the creation of users, and deletion of users in MongoDB through admin access and then to allow users to login via their credentials.

The website has 5 pages and one should follow this sequence.

website/admin -> login with
userID = 1234 || password = admin

Redirects to website/manageUsers

website/createUsers → Now create a user with any non-whitespace userID and password.

website/user → Now with the created credentials, log in.

Redirects website/userHome

website/userHome → Now At userHome you can look at user data and log out.

On logging out, ***Redirects to website/users***

3. Hosting: Describe the hosting environment and deployment process in a document.

To host the website, I used the **Railway** deployment platform.

Railway uses nixpacks to build a dockerfile from the GitHub repository that is hosted.

We set the node_version to v20 and nixpacks reads our repository specifically the package.json file.

Then it starts a docker image, installs all our dependencies and generates a dockerfile for this.

The dockerfile is then outputted to Railway which then uses it to run the image in its environment.

Then Railway runs the ***npm start*** command which starts the express instance.

Railway also lets us set environment variables which I ended up using to set a mongoDB_URL.

Railway also constantly listens to the GitHub repository. Any new pushes to the master branch rerun the entire build and deploy process.

4. Project Link: Provide a link to the application.

GitHub Repo : **Simple AUTH**

Live Demo : **Simple AUTH**