



### BACKLOGGER

Media consumption tracking platform, designed to keep users engaged while they progress through any and all type of media. The app streamlines the user's interaction with their media, simplifying the decision-making process and increasing the time spent engaging with the content they love.



### TECHNOLOGICAL STACK

#### Front-end



#### Back-end



In pursuit of streamlining the development process we had to choose our technologies based on the team's experience and opted out of using a framework and instead went for a direct implementation of HTML, CSS and JavaScript for the frontend. Our choice of Node.js and Express.js was motivated by desire of having some overlap in coding language between the frontend and backend, while MongoDB was selected due to the team's prior experience with it. This intentional simplicity lays the groundwork for easy future enhancements and contributions.

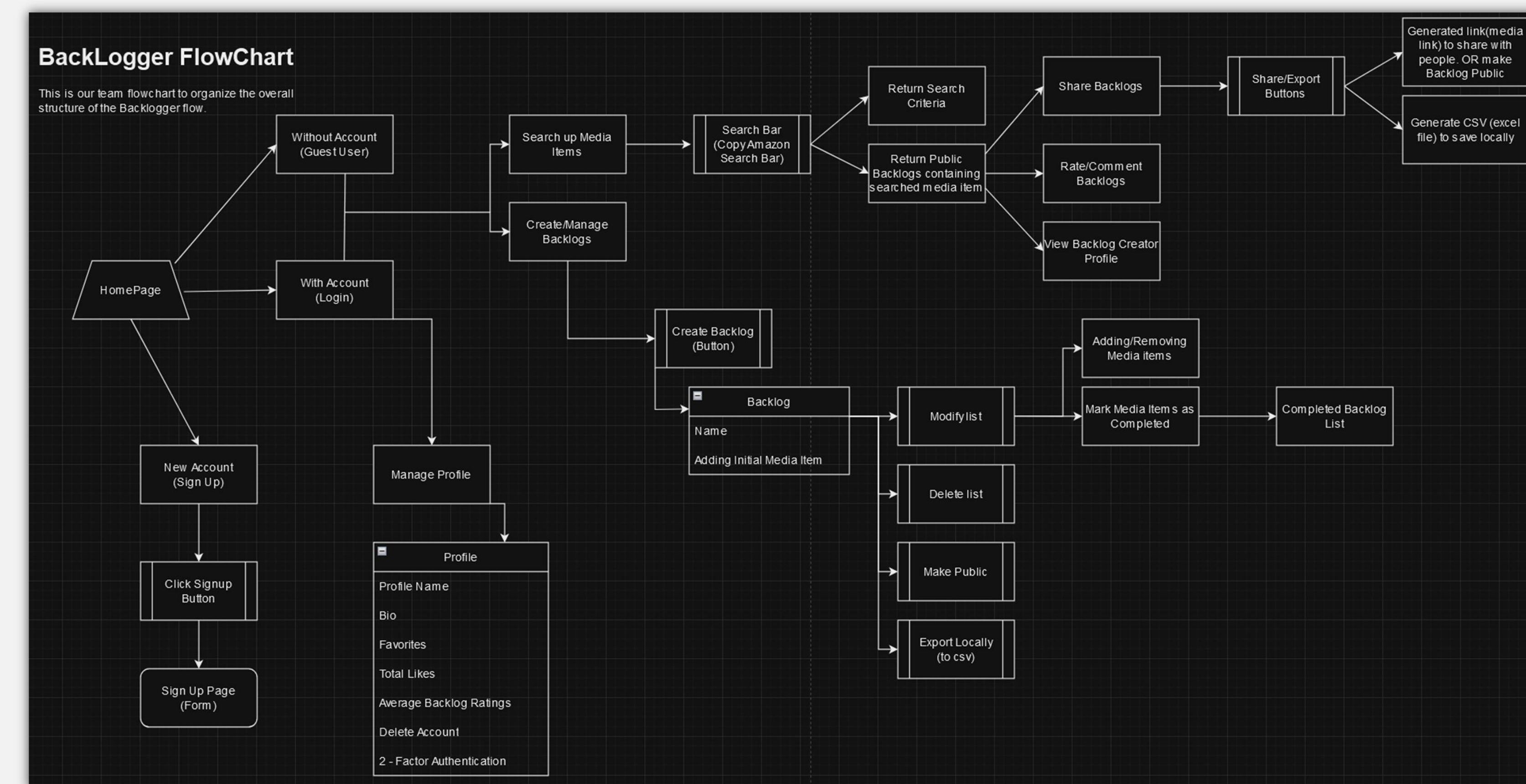
### CURRENT SYSTEM

Nowadays there are multiple forms of entertainment and media that are always trying to catch our attention. It can be difficult to keep track of what you are watching across several different apps and services, and making a choice can be an overwhelming task; that is where BackLogger comes in. BackLogger is a simple and fun way to keep track of all the media you are consuming, as well as being able to share, rate and comment on your favorite shows, movies, games, etc.

### BACKLOGGER FUNCTIONS

BackLogger's functional requirements are:

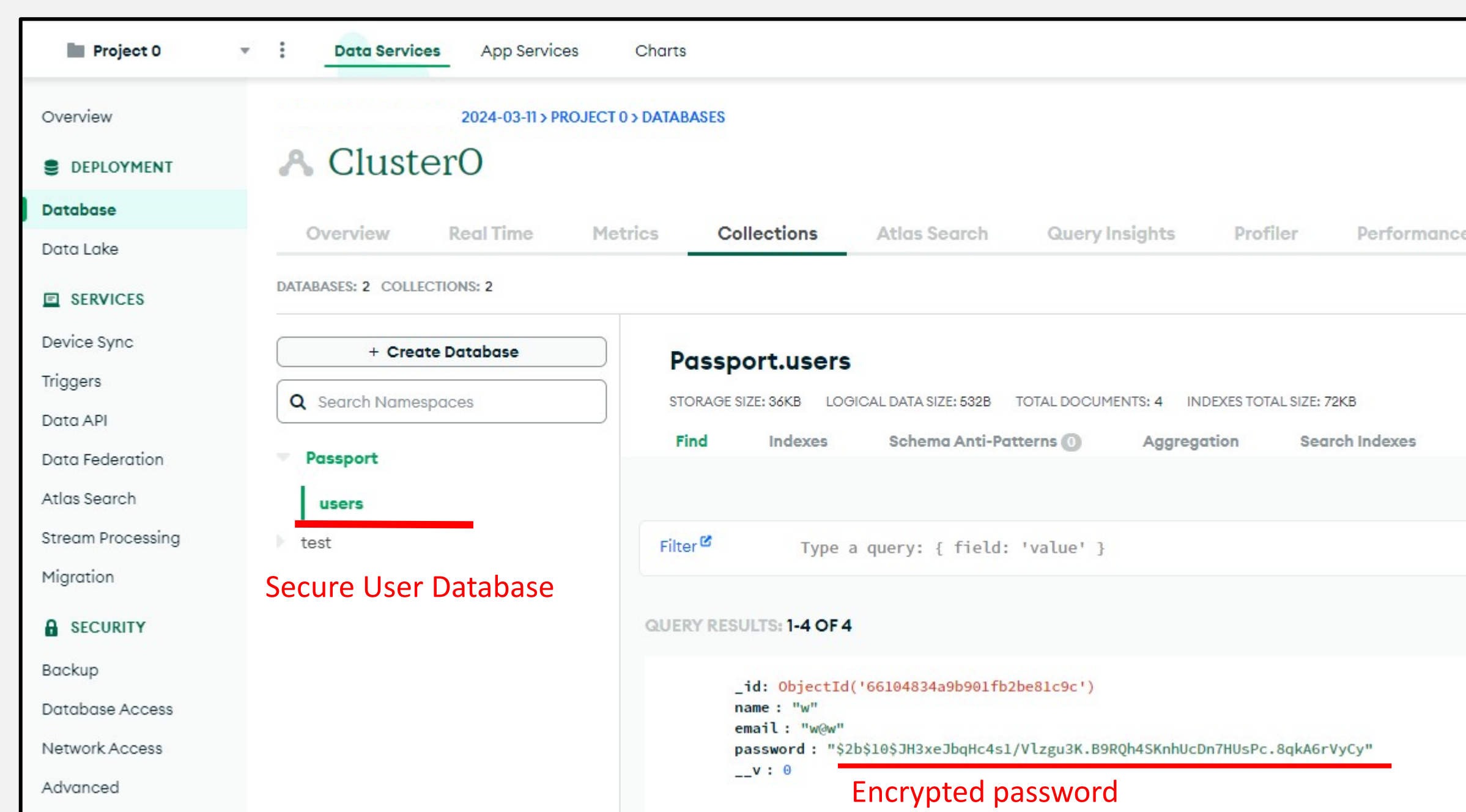
- User account creation/management
- List interface for creation, editing and deletion
- Features to mark progress of media items
- Search functionalities
- Import/Export list



### BACKEND IMPLEMENTATION

As a backend developer, I Focused on the main functionality of the webapp. We worked on designed database structure to be used in Json format. Then we worked on creating different api calls to gather data and save data to the database. We worked with Technologies of node.js and MongoDB. We programmed in a language called JavaScript.

### ENCRYPTED USER DATABASE



### PASSPORT



### USER AUTHENTICATION

I focused on the user authentication of the webapp with the use of the passport.js package.

Passport is a simple authentication middleware with the whole purpose of authenticating request. Passport does not require any specific routes or schema which simplifies its use while connecting to a database.

The backend team worked on a User Schema and created a secure and flexible database with the use of passport, bcrypt and cookie management methods.

### IMPLEMENTATION: Agile Development Methodology

- **Sprint Execution:** Over 7 sprints, the team and I advanced BackLogger's capabilities, each cycle introducing new, robust features.
- **Daily Scrum Meetings:** Contributed our team in daily scrums to swiftly address challenges and maintain momentum towards our collective goals.
- **Backlog Management:** Ensured our backlog accurately represented the project's evolving needs, keeping tasks targeted and efficient.
- **Sprint Reviews & Learning:** Contributed with sprint reviews and retrospectives, fostering a culture of continuous improvement and teamwork.
- **Integration & User Experience:** Worked closely with the team to integrate back-end and front-end modules, focusing on a responsive design for a better user experience.
- **User Feedback Implementation:** After development, we collaboratively fine-tuned UI/UX, implementing user feedback to improve media tracking.
- **Launch Readiness:** Together, we prepared for the final sprint, focusing on thorough documentation and a polished presentation of BackLogger.

### ACKNOWLEDGEMENT

The material presented in this poster is based upon the work supported by Dr. Masoud Sadjadi. We/I thank Dr. Masoud Sadjadi for his/her/their assistance and mentorship that I/we received throughout the senior design project.