**Data Flow Diagram (DFD) for Registration App**

**Registration App** allows users to **sign up, log in, and manage authentication** using **React (frontend) and Express with MongoDB (backend).** Below is the **DFD representation:**

**Level 0 (Context Diagram)**

At the highest level, the system consists of:

1. **External Entity: User** → Registers an account and logs in.
2. **Process: Registration & Authentication System** → Handles user input, validation, and authentication.
3. **Data Store: MongoDB User Database** → Stores user credentials securely.

+--------------------+

| External Entity: |

| User |

+--------------------+

|

v

+--------------------+

| Registration & | (Process 1.0)

| Authentication |

| System |

+--------------------+

|

v

+--------------------+

| MongoDB User | (Stores user credentials)

| Database |

+--------------------+

**Level 1 DFD (Decomposed Processes)**

Breaking down **Process 1.0 (Registration & Authentication System)** into detailed steps:

+--------------------+ +------------------------+

| External Entity: | | Process 1.1 - Input |

| User | -----> | (Enter Registration |

| | | or Login Details) |

+--------------------+ +------------------------+

| |

v v

+--------------------+ +------------------------+

| Process 1.2 - | | Process 1.3 - Encrypt |

| Validate Input | -----> | (Hash Password) |

+--------------------+ +------------------------+

| |

v v

+--------------------+ +------------------------+

| Process 1.4 - | | Process 1.5 - Verify |

| Store User Data | -----> | (Check User Exists) |

+--------------------+ +------------------------+

| |

v v

+--------------------+ +------------------------+

| Process 1.6 - | | Process 1.7 - Return |

| Authenticate User| -----> | (Send Response to UI) |

+--------------------+ +------------------------+

**Explanation of Level 1 DFD Processes**

1. **Process 1.1 - User Enters Registration or Login Details**
   * The user types their name, email, and password in the **Sign-Up** or **Login** form.
2. **Process 1.2 - Validate Input**
   * The system checks if all required fields are filled and verifies the email format.
3. **Process 1.3 - Encrypt Password**
   * The system **hashes** the password using bcrypt.js before storing it.
4. **Process 1.4 - Store User Data (MongoDB)**
   * If registering, the **new user data** is saved in the database.
   * If logging in, the system **retrieves** user data from the database.
5. **Process 1.5 - Verify User**
   * The system checks if the **email already exists** (during sign-up).
   * The system checks if the **password matches the stored hash** (during login).
6. **Process 1.6 - Authenticate User**
   * If login is successful, the system generates a **JWT Token** for session management.
7. **Process 1.7 - Return Response to UI**
   * If successful, the system sends a **success message** or redirects the user.
   * If unsuccessful, an **error message** is displayed (e.g., "Incorrect password").

**Data Flow Summary**

✔ **User registers/logs in → System validates & encrypts data → Stores/retrieves from database → Authenticates user → Sends response**  
✔ Uses **React (useForm, API calls) + Express.js + MongoDB** for full-stack implementation.  
✔ **Security** with **bcrypt.js (password hashing)** and **JWT (session authentication).**