**Within Project Documentation**

**Team**

* Yassin Mohammad Bedier
* Islam Akram Abdelmotaleb
* Ahmed Mohamed Khattab
* Ahmed Hatem Roshdy
* Ibrahim Hazem Labib

**Project Overview**

Within is a single-page-application, built with the MERN Stack. It serves as an internal organization portal, that could be used for universities, companies, etc. Currently, it provides user authentication and dashboard functionality. It includes features like user registration, login, password reset, and a dynamic dashboard with announcements, posts, and calendar integration. This is the beta version so it does not have all functions that we have in mind, but for the final project there will be much more features.

**Technology Stack (MERN)**

**Backend**

* Node.js
* Express.js
* MongoDB (with Mongoose)
* bcryptjs (for password hashing)
* CORS
* Body-parser

**Frontend**

* React.js
* React Router DOM
* React Bootstrap
* Axios
* React Calendar

**Project Structure**

within/

├── client/

│ └── src/

│ ├── components/

│ │ ├── boxes.js

│ │ ├── forgetPassword.js

│ │ ├── header.js

│ │ ├── login.js

│ │ ├── register.js

│ │ └── sidebar.js

│ ├── css/

│ │ ├── App.css

│ │ ├── Boxes.css

│ │ ├── Calendar.css

│ │ ├── Header.css

│ │ ├── index.css

│ ├── App.js

│ ├── Route.js

│ └── index.js

└── server/

├── models/

│ └── Users.js

├── index.js

└── package.json

**MongoDB Cluster Structure**

**Users**

* \_id: int32 *(unique)*
* username: string *(unique)*
* email: string *(unique)*
* password: string *(hashed)*
* type: string *(user | admin)*

**Server-Side**

**API Endpoints**

**1. POST /login**

- Purpose: Authenticate user

- Request Body:

{

"username": "string",

"password": "string"

}

- Response: { success: boolean, message: string, type: string }

**2. POST /register**

- Purpose: Create new user

- Request Body:

{

"username": "string",

"email": "string",

"password": "string",

"type": "string"

}

- Response: { success: boolean, message: string }

**3. POST /forgetPassword**

- Purpose: Reset user password

- Request Body:

{

"username": "string",

"password": "string",

"newPassword": "string"

}

- Response: { success: boolean, message: string }

**Client-Side**

**Components**

**1. Login Component**

* Handles user authentication
* Form fields: username, password
* Navigation to register and forgot password pages

**2. Register Component**

* Handles user registration
* Form fields: username, email, password
* Input validation for email format

**3. ForgetPassword Component**

* Handles password reset functionality
* Form fields: username, current password, new password

**4. Dashboard Components**

1. Header: Search functionality and notifications
2. Sidebar: Navigation menu and welcome message
3. Boxes: Grid layout with announcements, posts, and calendar

**Routing**

<BrowserRouter>

<Routes>

<Route index element={<Login />} />

<Route path="login" element={<Login />} />

<Route path="register" element={<Register />} />

<Route path="reset" element={<ForgetPassword />} />

<Route path="home" element={<App />} />

</Routes>

</BrowserRouter>

**Setup Instructions**

**Server Setup**

**1. Navigate to server directory:**

cd server

**2. Install dependencies:**

npm install

**3. Start server:**

node index.js

**Client Setup**

**1. Navigate to client directory:**

cd client/client

**2. Install dependencies:**

npm install

**3. Start development server:**

npm start

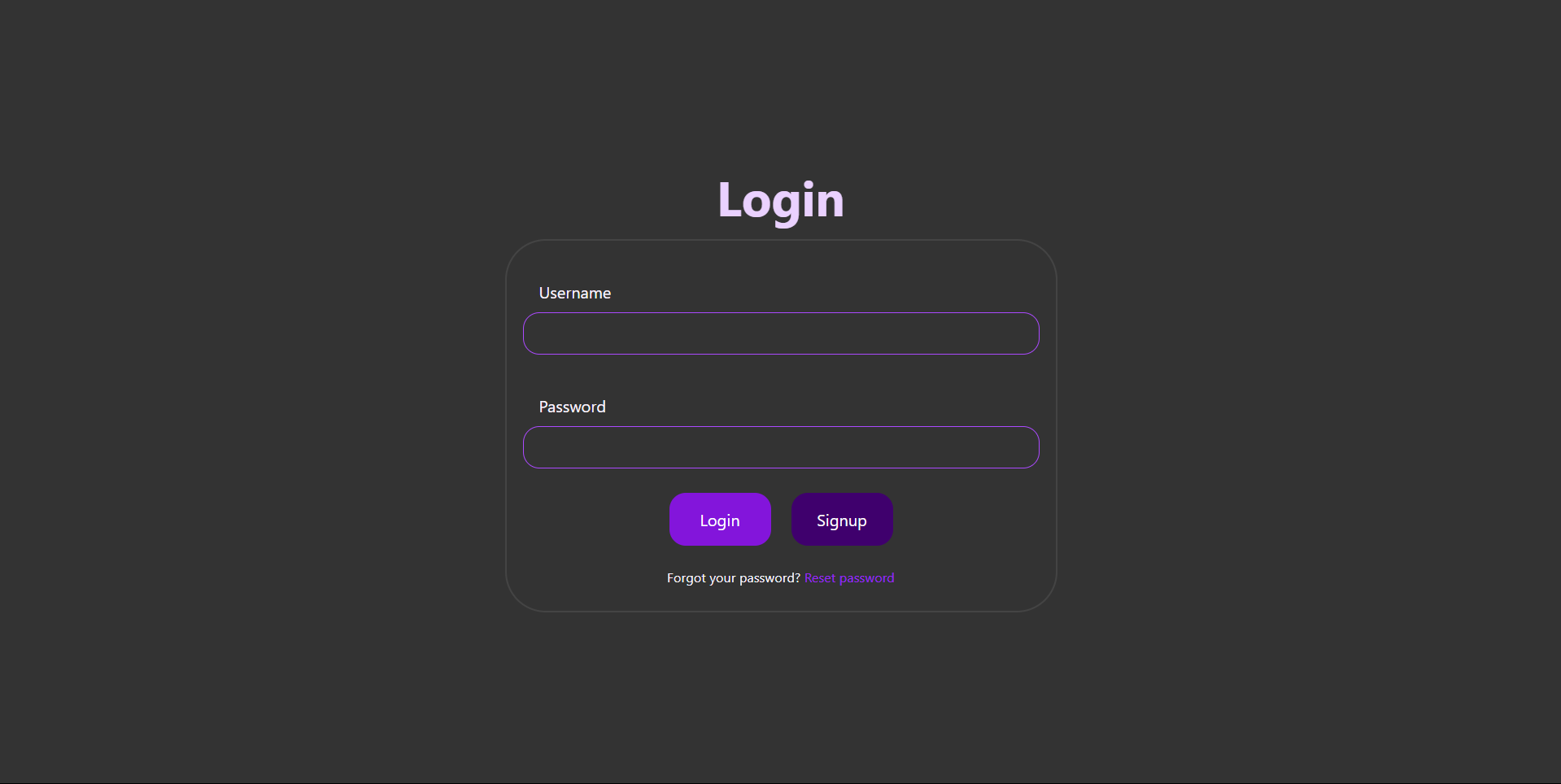
**Environment Requirements**

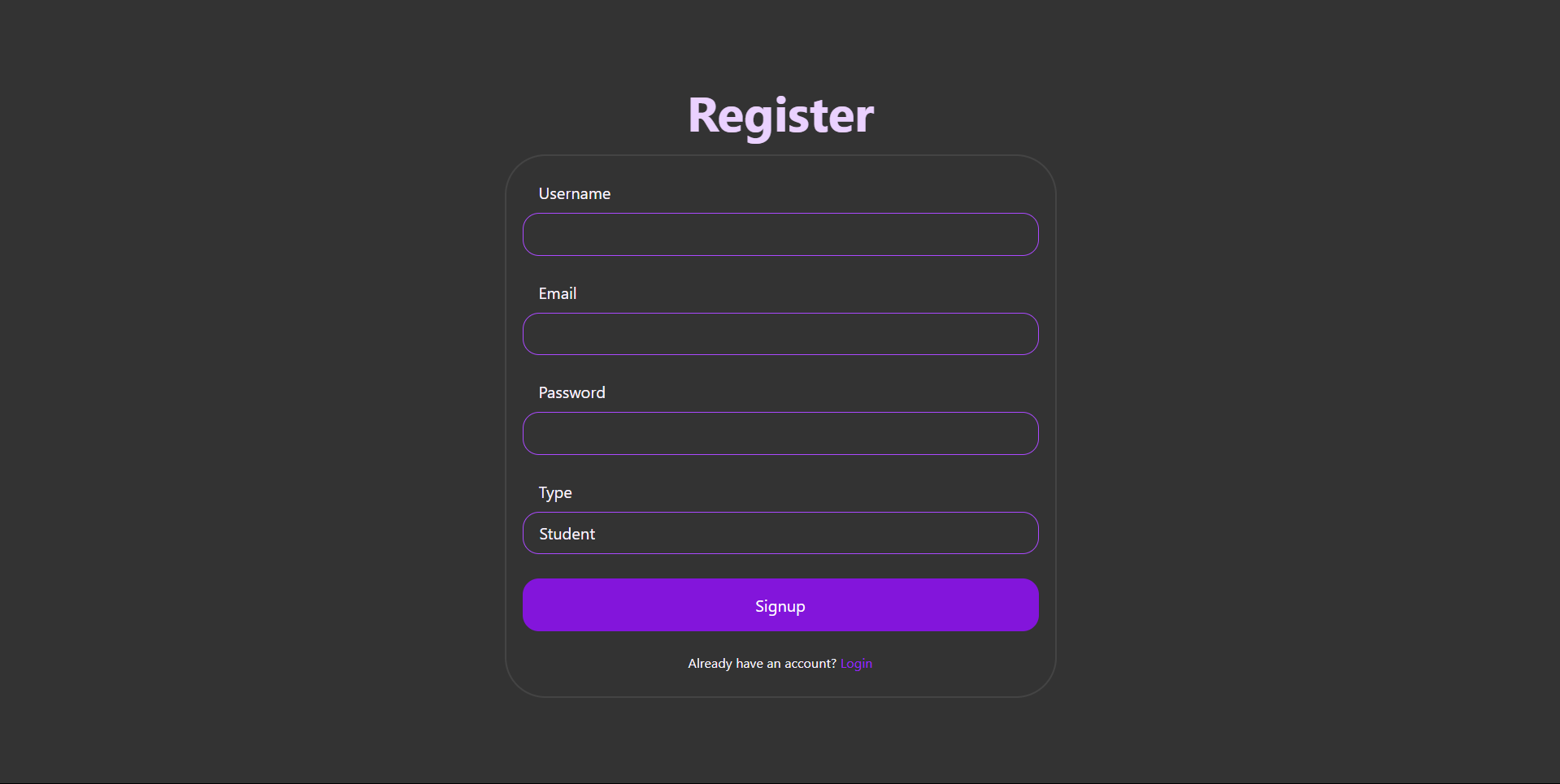
* Node.js version: 14.x or higher
* MongoDB connection string required
* Port 4000 for server
* Port 3000 for client (default React port)

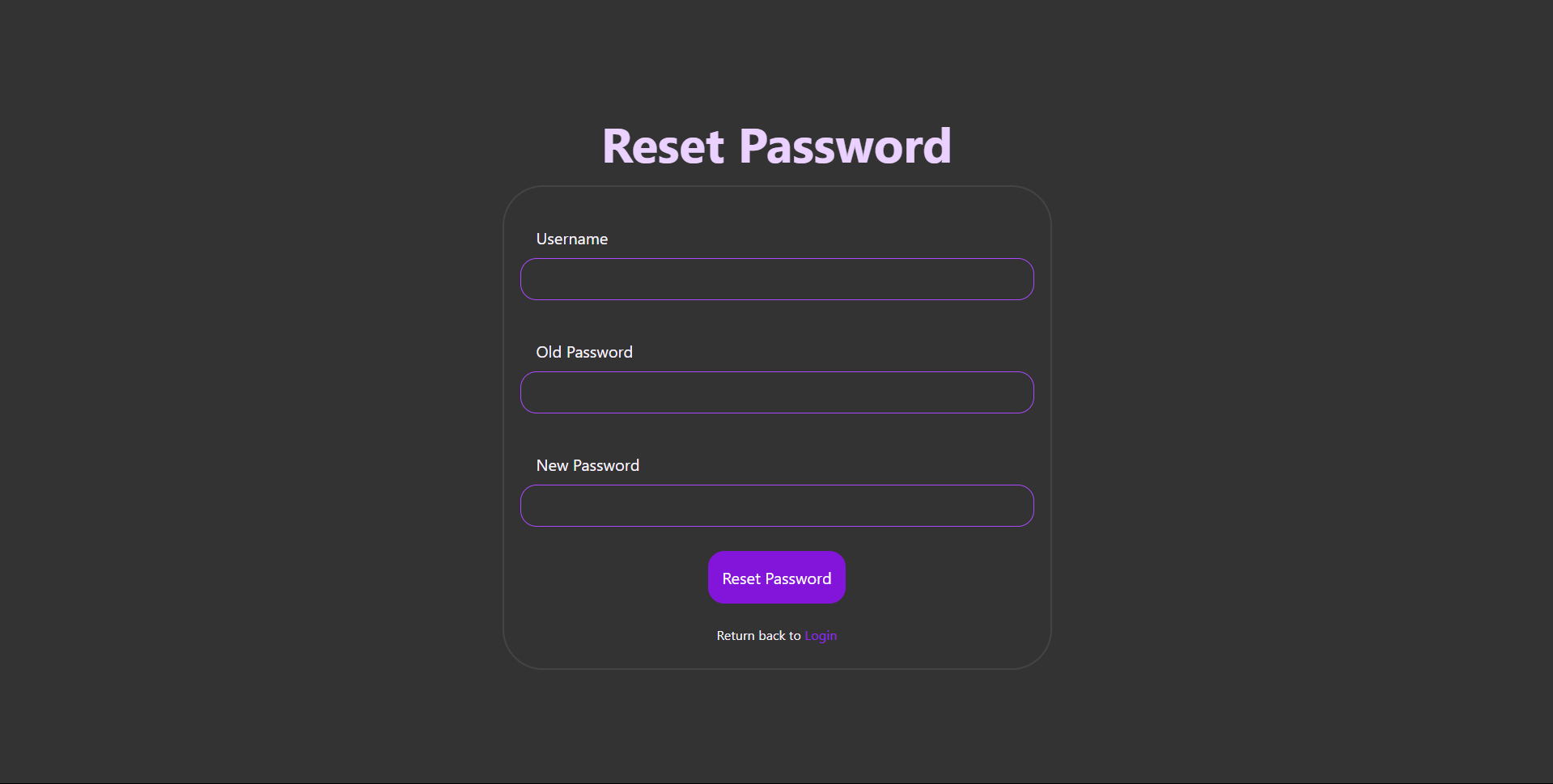
**Security Features**

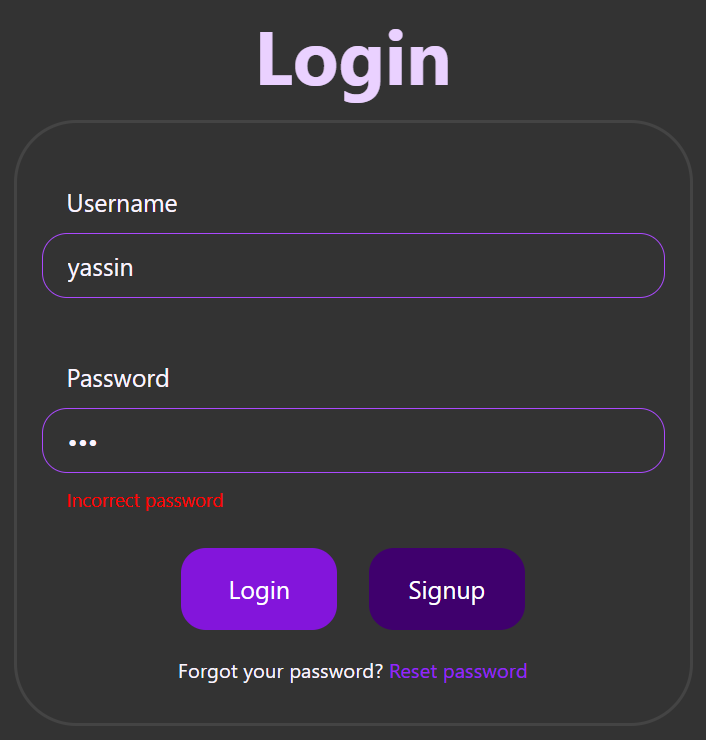
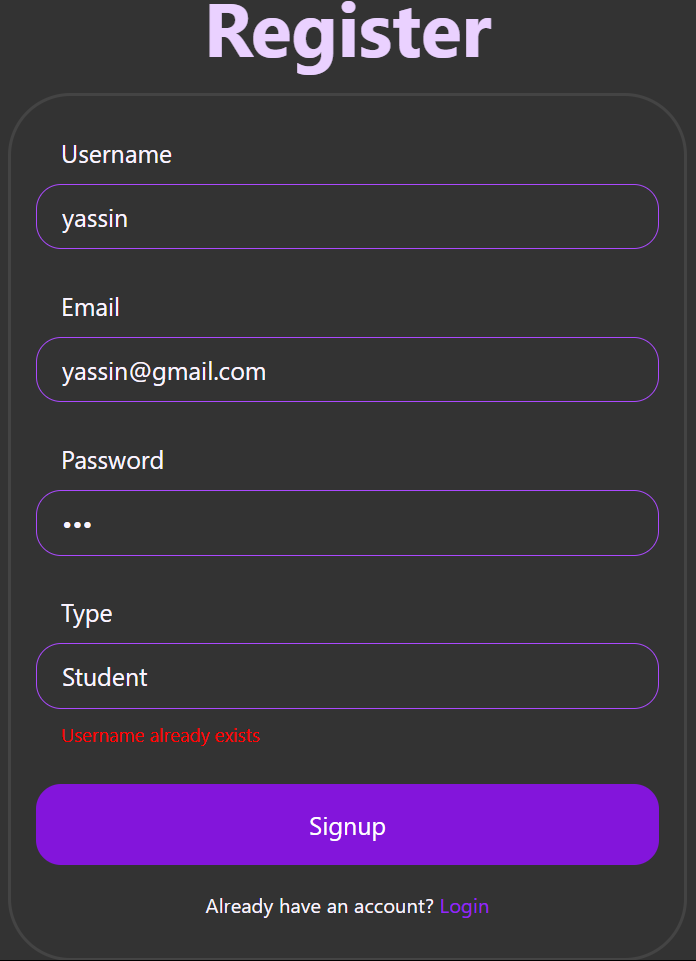
* Password hashing using bcryptjs
* Case-insensitive username/email handling
* Input validation for registration
* Error handling and user feedback

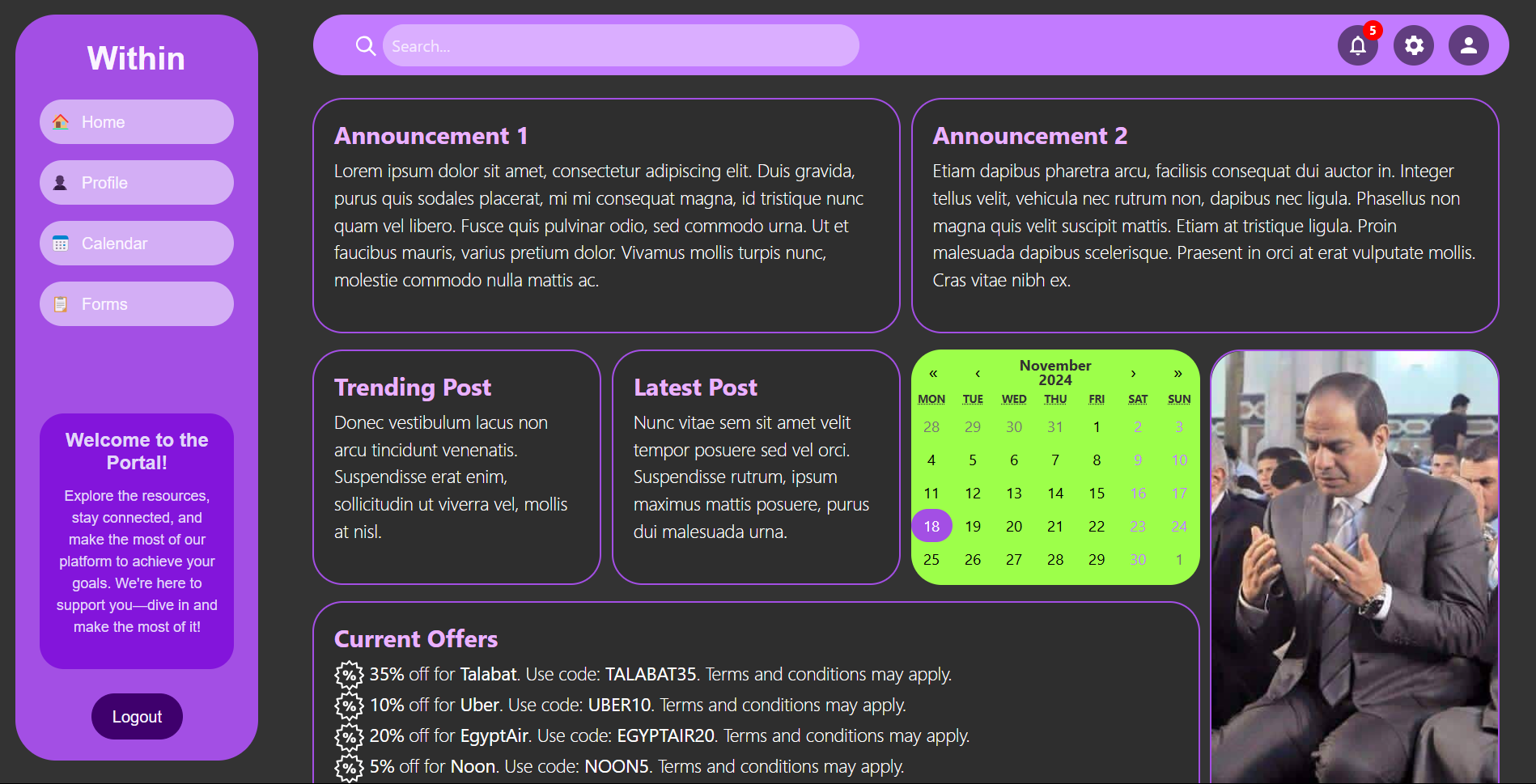
**Screenshots**

**Login**

**Regsiter**

**Reset Password**

**Errors Preview**

**Main Dashboard**