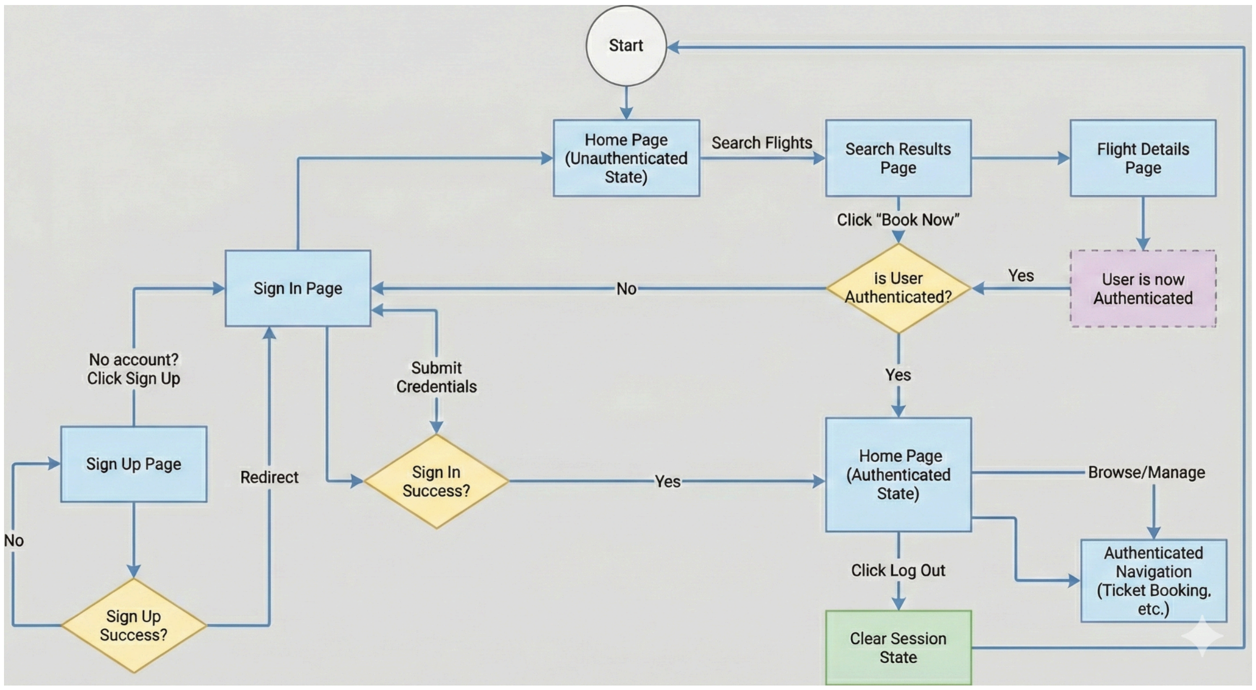
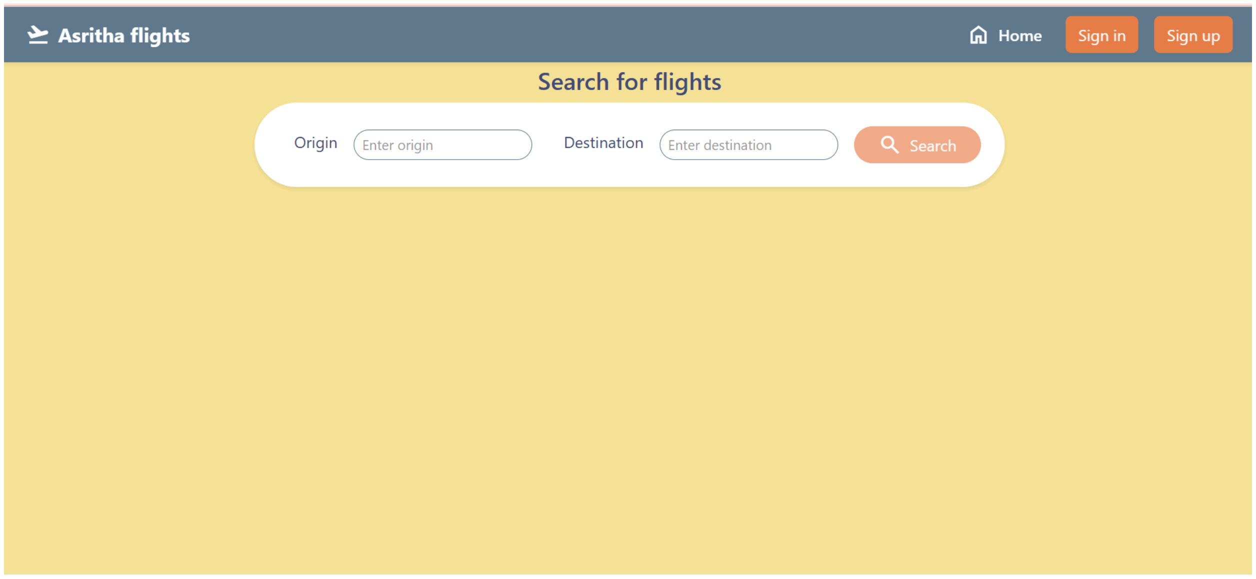
Flow diagram:

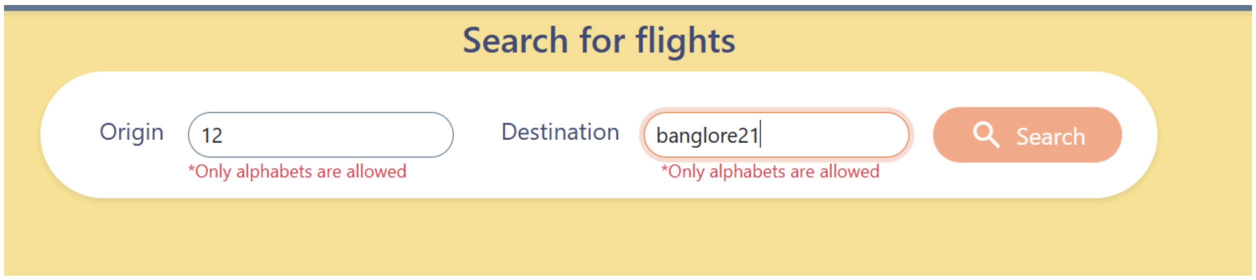


Home Page:



Validation for search form:

Search button disabled until form is valid



12

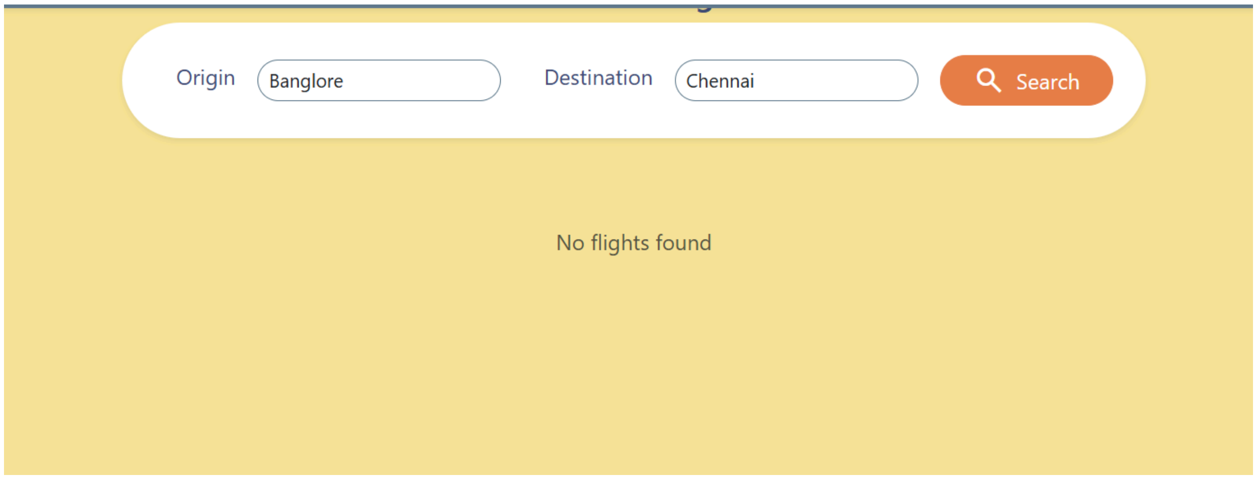
*Only alphabets are allowed

banglore21

*Only alphabets are allowed

Searching Flights:

If no flights found:

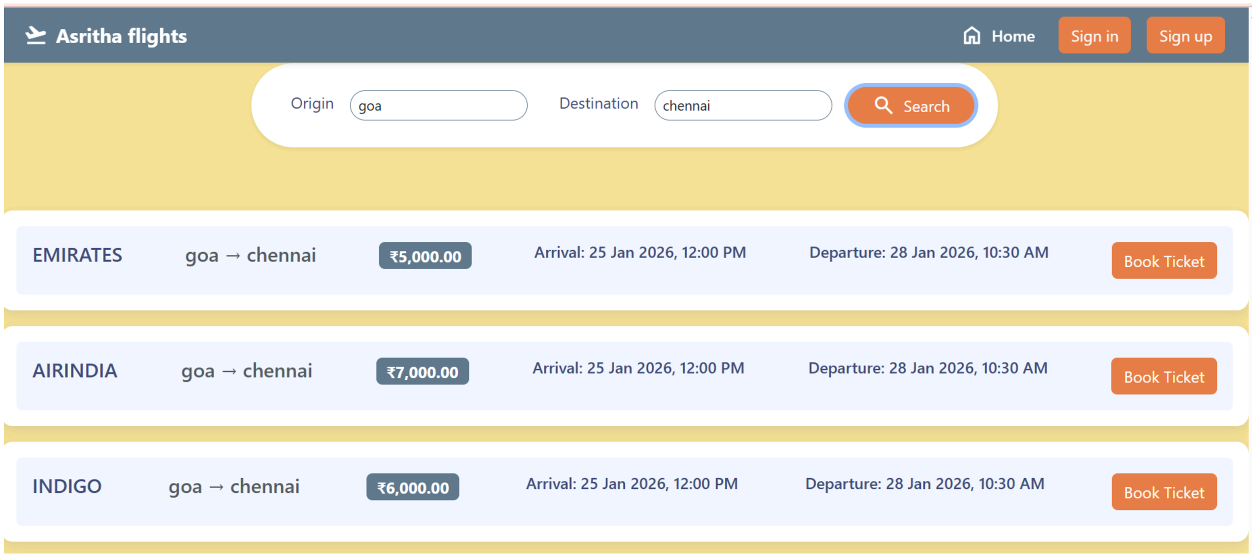


Banglore

Chennai

No flights found

If flights found:

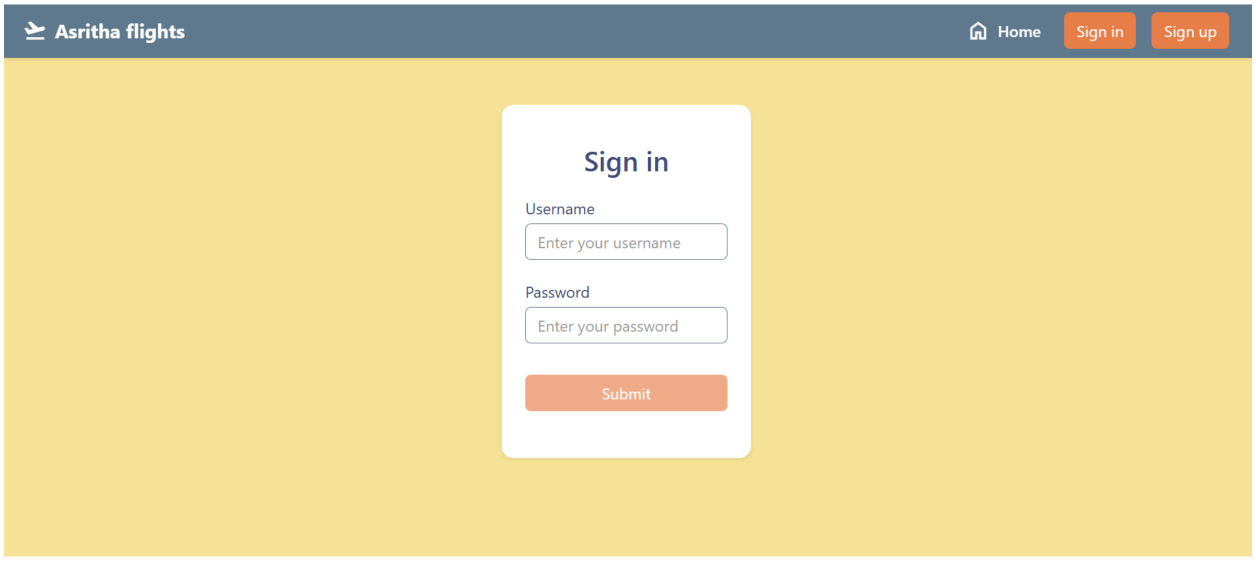


EMIRATES	goa → chennai	₹5,000.00	Arrival: 25 Jan 2026, 12:00 PM	Departure: 28 Jan 2026, 10:30 AM	Book Ticket
AIRINDIA	goa → chennai	₹7,000.00	Arrival: 25 Jan 2026, 12:00 PM	Departure: 28 Jan 2026, 10:30 AM	Book Ticket
INDIGO	goa → chennai	₹6,000.00	Arrival: 25 Jan 2026, 12:00 PM	Departure: 28 Jan 2026, 10:30 AM	Book Ticket

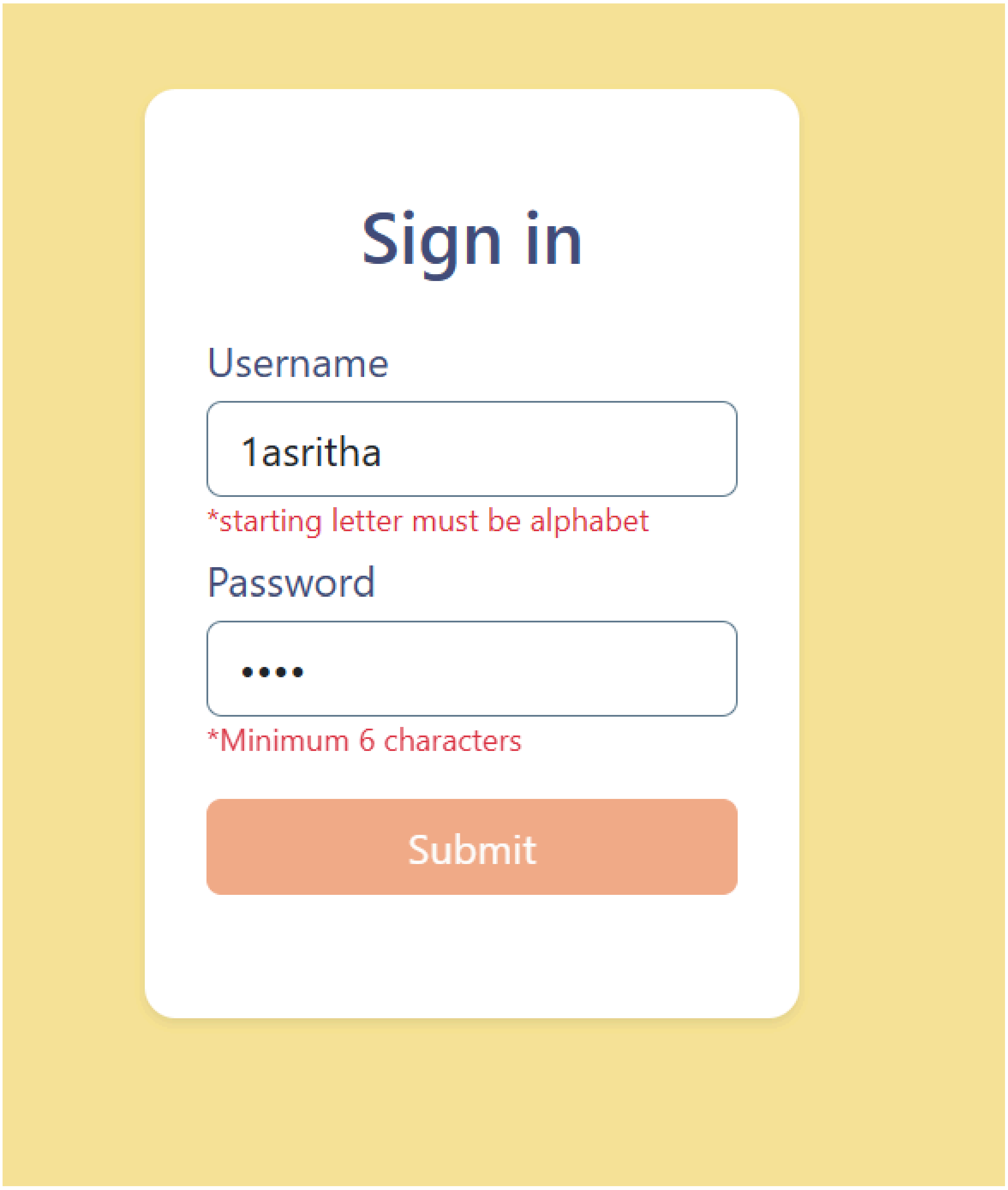
Book Ticket button:

Redirects to sign in page when clicked and not signed in

Sign in page:


Asritha flights
Home
Sign in
Sign up
Sign in
Username
Enter your username
Password
Enter your password
Submit

Validations:


Sign in

Username

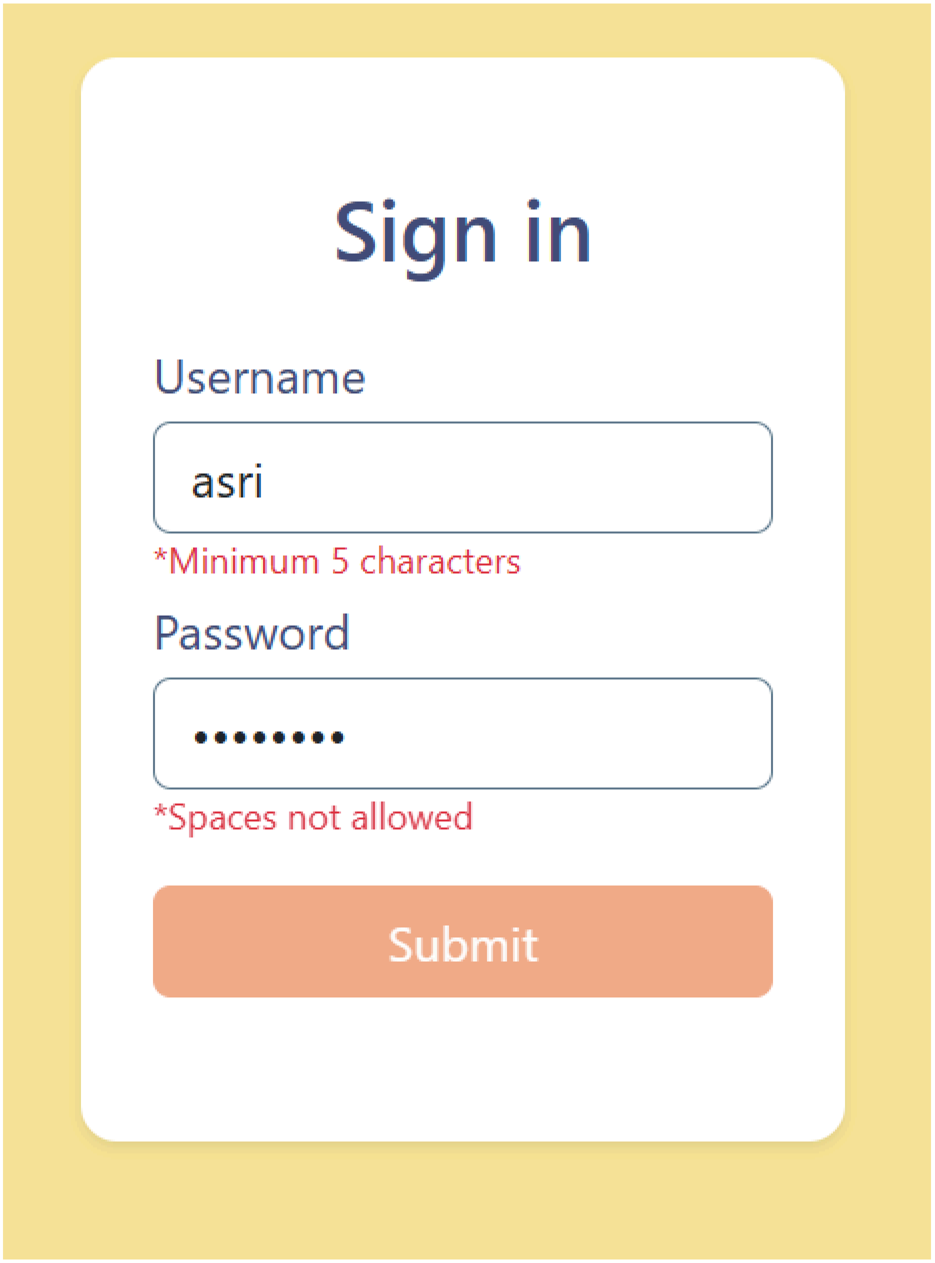
1asritha

*starting letter must be alphabet

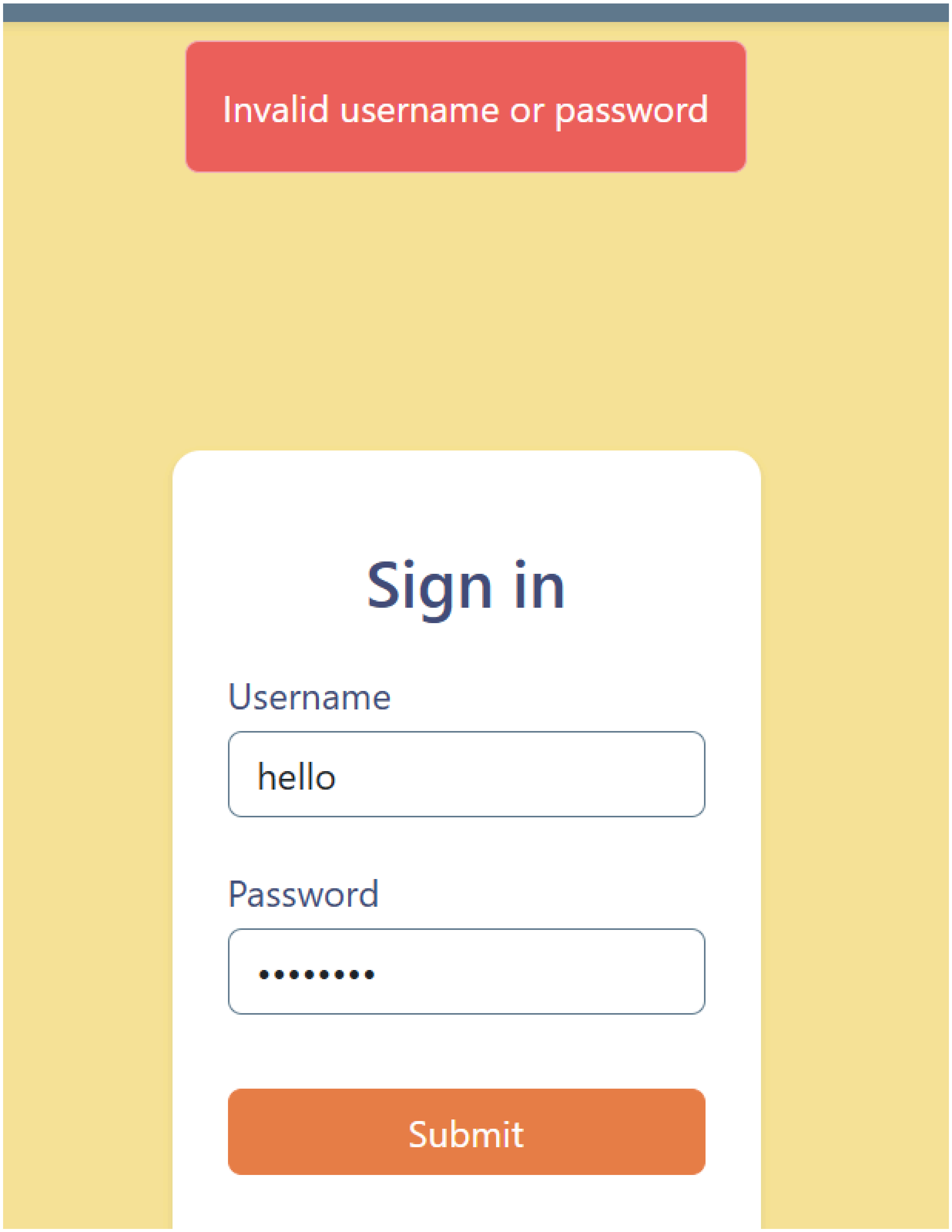
Password

••••

*Minimum 6 characters



If wrong username or password:


Invalid username or password

Sign in

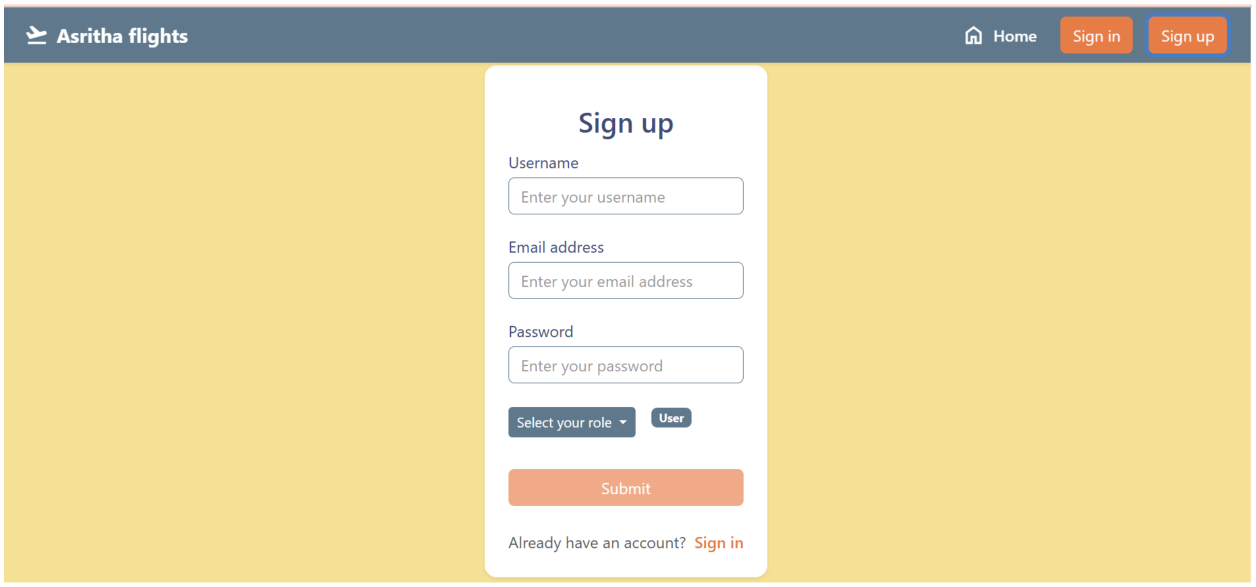
Username

hello

Password

•••••••••

Sign up page:



Sign up

Username

Enter your username

Email address

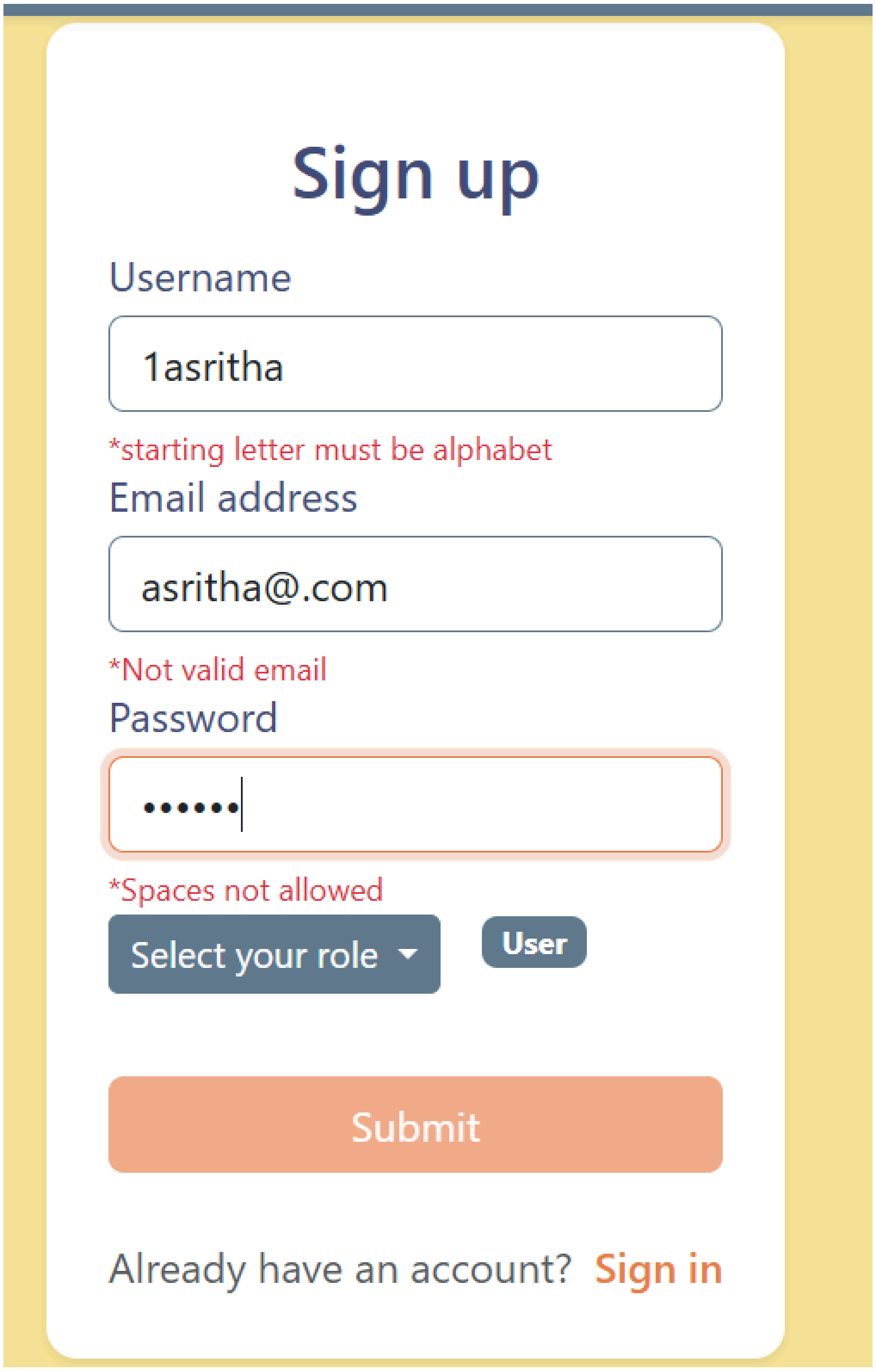
Enter your email address

Password

Enter your password

Already have an account? Sign in

Validations handled:



Sign up

Username

1asritha

*starting letter must be alphabet

Email address

asritha@.com

*Not valid email

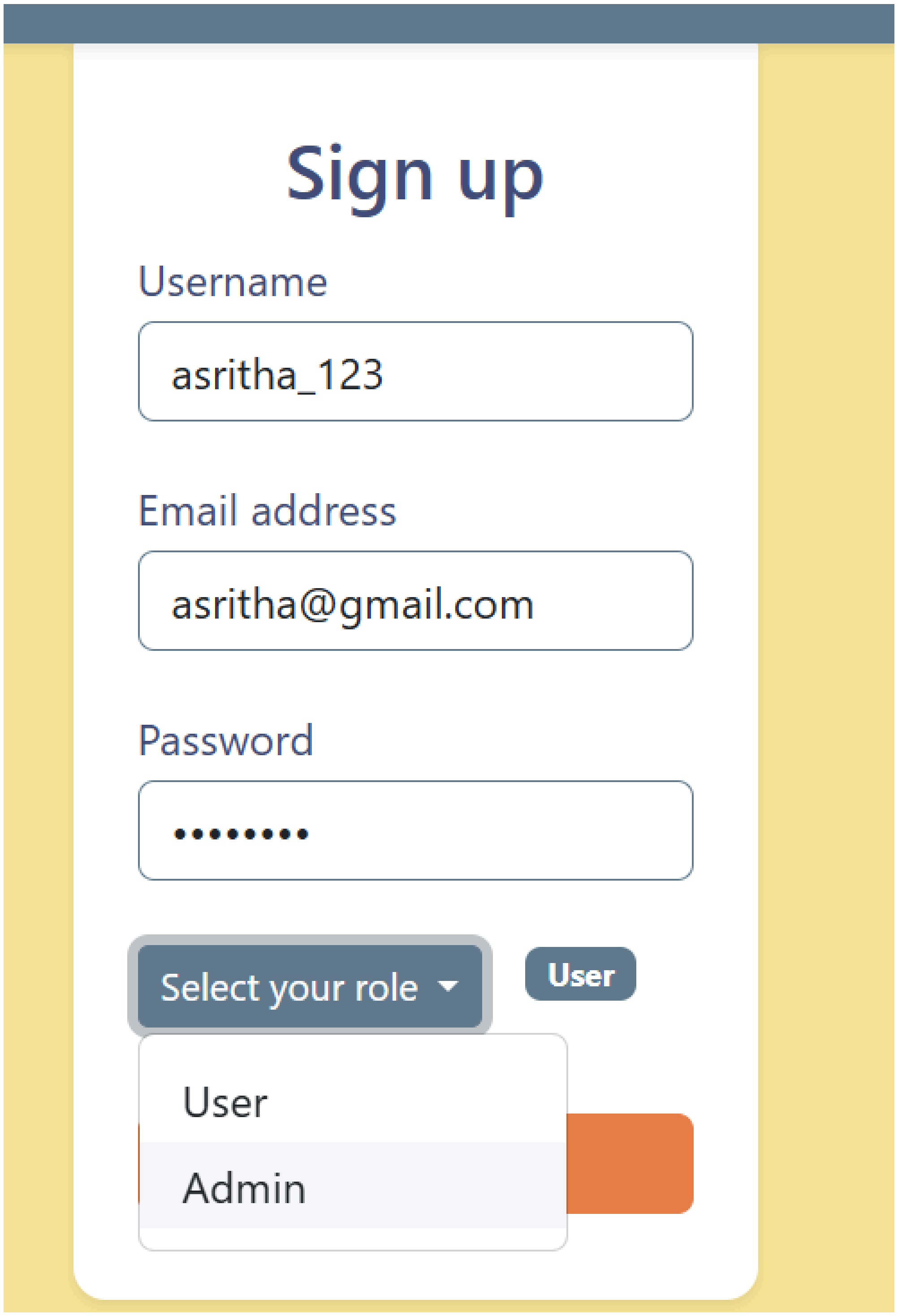
Password

••••••

*Spaces not allowed

Already have an account? Sign in

Selection of roles in drop down button:


Sign up

Username

asritha_123

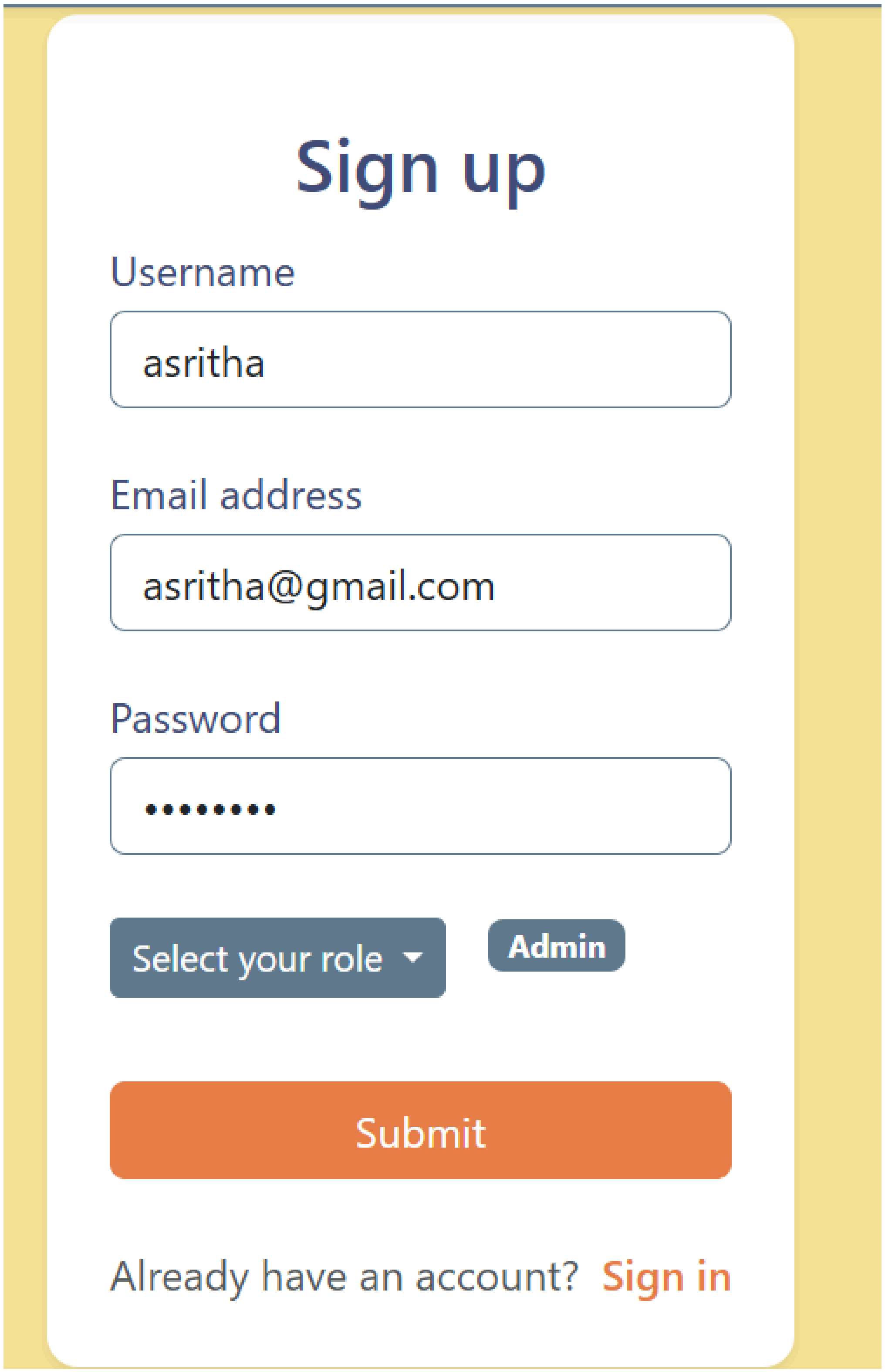
Email address

asritha@gmail.com

Password

•••••••••

Submit button enabled only after form is valid



Sign up

Username

asritha

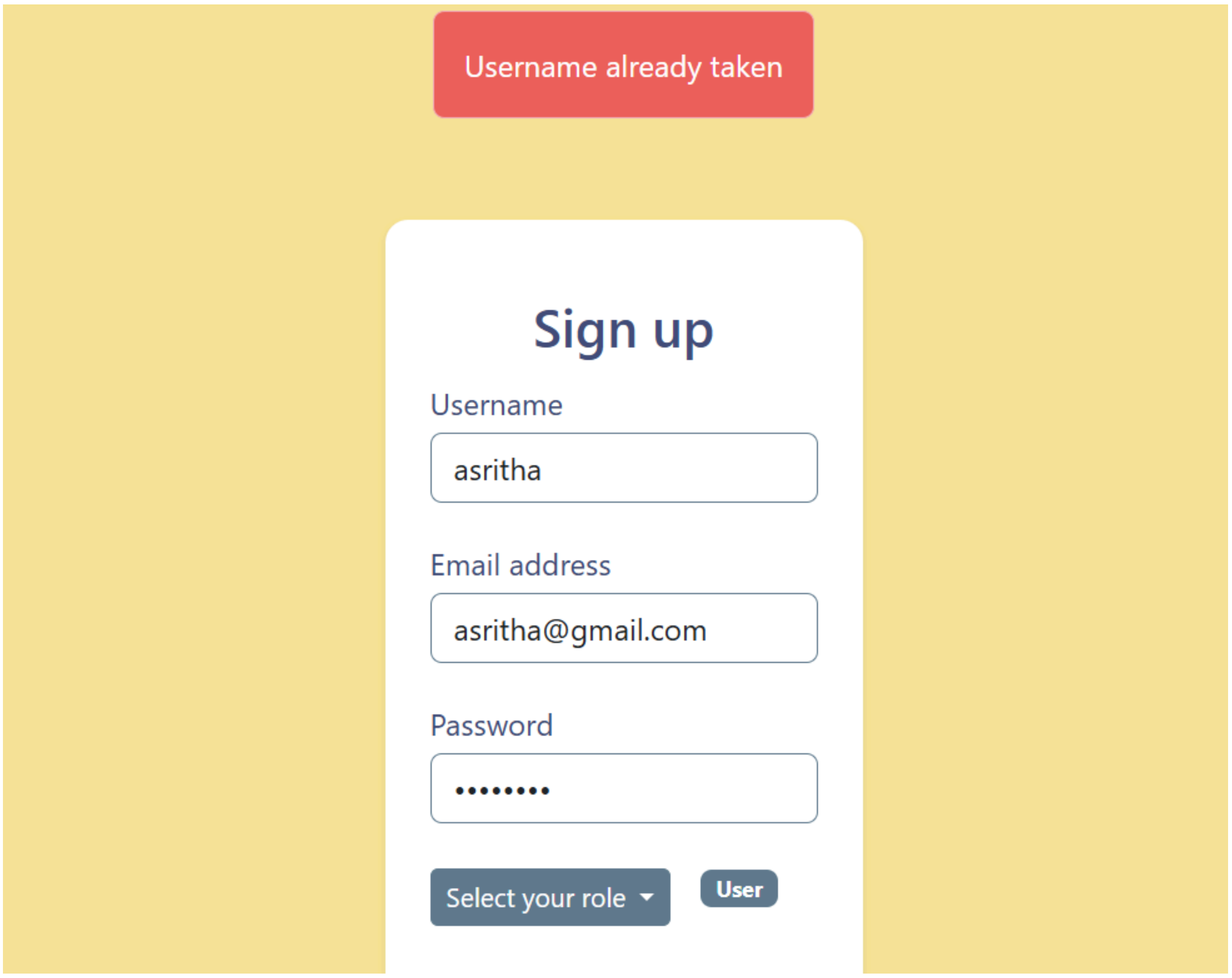
Email address

asritha@gmail.com

Password

•••••••••

Showing if user name already exists:



Username already taken

Sign up

Username

asritha

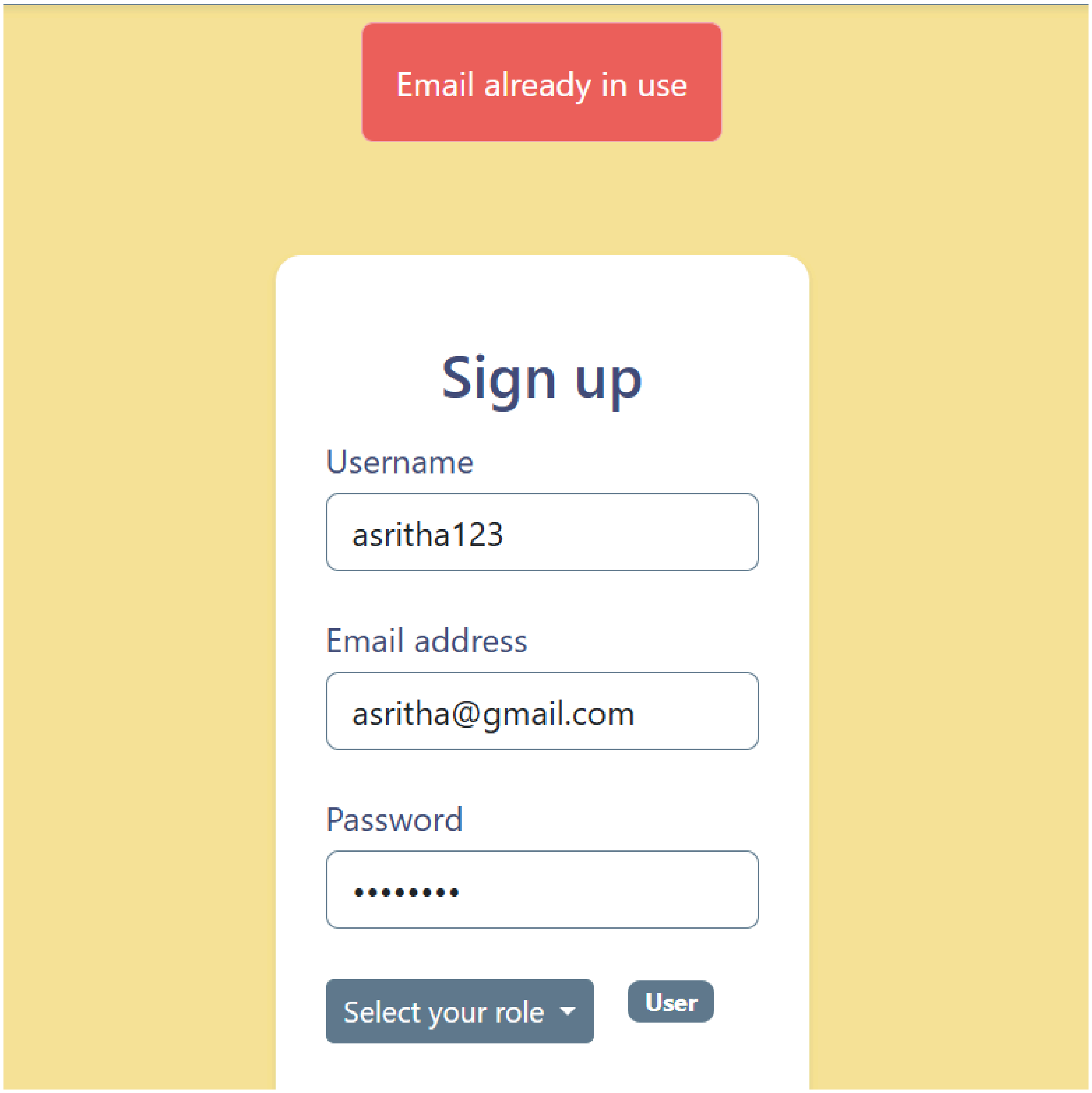
Email address

asritha@gmail.com

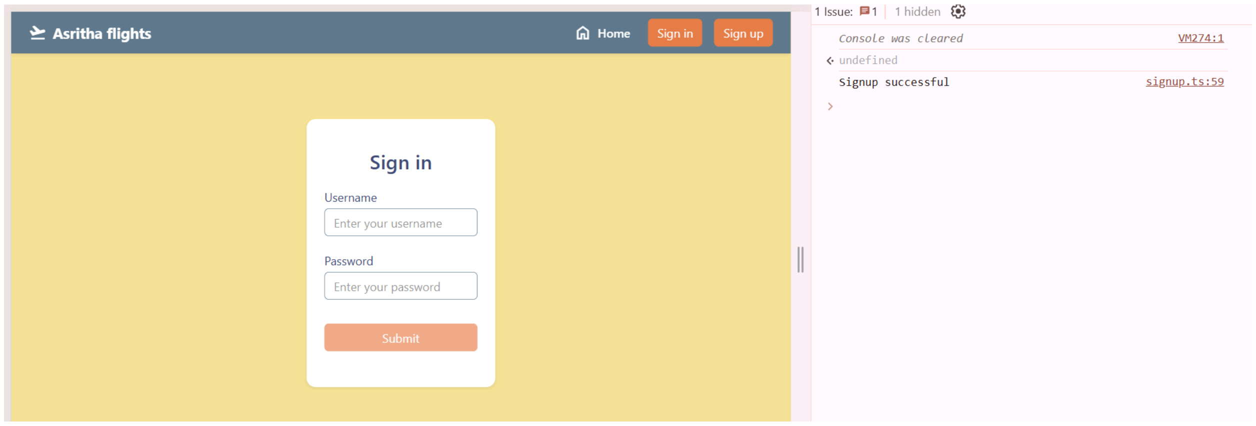
Password

•••••••••

Showing if email already in use:

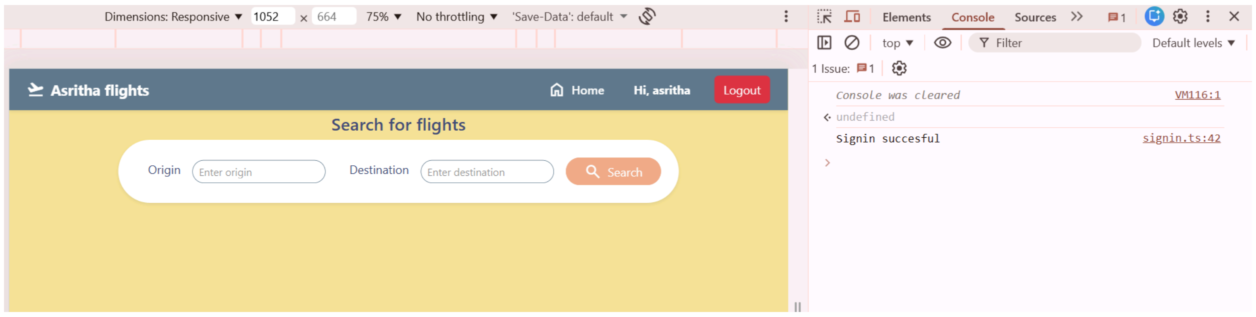


Sign up redirects to sign in



```
Console was cleared                                   VM274:1
<· undefined
Signup successful                                signup.ts:59
>
```

Sign in redirects to home page with changes in nav bar

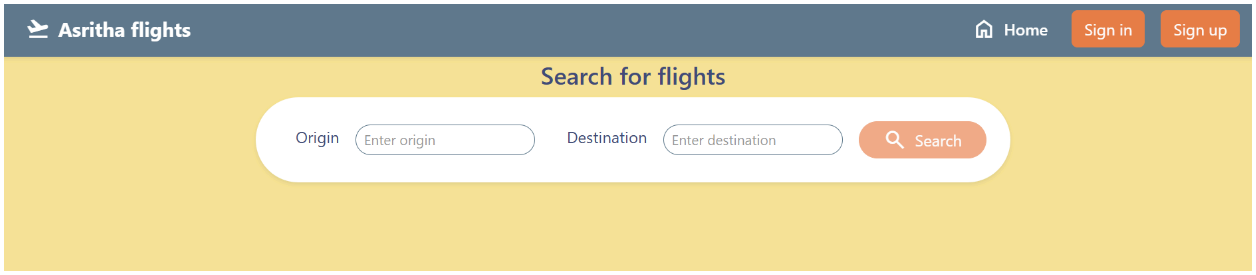


```
Console was cleared                                   VM116:1
<· undefined
Signin succesful                                  signin.ts:42
>
```

Changes in nav bar.

Logout button added

After logging out nav bar changes back and page refreshes:



Routing

Configuration: `src/app/app.routes.ts`

Defined Routes

- **`/` → Home**
  `src/app/components/home/home.ts`
- **`/signup` → Signup**
  `src/app/components/signup/signup.ts`
- **`/signin` → Signin**
  `src/app/components/signin/signin.ts`
- **`/flights` → FlightList**
  `src/app/components/flight-list/flight-list.ts`
- **`/book` → TicketBooking** *(protected by `userOrAdminGuard`)*
  `src/app/components/ticket-booking/ticket-booking.ts`
- **`/register` → PassengerRegistration** *(protected by `userOrAdminGuard`)*
  `src/app/components/passenger-registration/passenger-registration.ts`

Used Bootstrap for css and overrided for customization

Route Guards

All guards are **reactive** and depend on the `currentUser` observable from `AuthService`. Redirections are handled using Angular `Router`.

`authGuard`

- **Source**: `src/app/gaurds/auth.gaurd.ts`
- **Logic**:
  - Allows navigation if a user is logged in.
  - Otherwise, redirects to `/signin`.
- **Status**: Implemented but **not currently applied** to any route.

`adminGuard`

- **Source**: `src/app/gaurds/admin.gaurd.ts`
- **Logic**:
  - Allows navigation only if `currentUser.roles` contains `ROLE_ADMIN`.
  - Otherwise, redirects to `/signin`.
- **Status**: Implemented but **not currently applied** to any route.

`userOrAdminGuard`

- **Source**: `src/app/gaurds/userOrAdminGuard.ts`
- **Logic**:
  - If no authenticated user → redirect to `/` and deny access.
  - If roles include `ROLE_USER` or `ROLE_ADMIN` → allow access.
  - Otherwise → redirect to `/` and deny access.
- **Usage**:
  - Protects `/book`
  - Protects `/register`

Navigation Flows

Navbar & Logout

- **Sources**:
  - `src/app/app.ts`
  - `src/app/app.html`

Behavior:

- When no user is authenticated:
  - Displays **Sign in** and **Sign up** buttons.
- When authenticated:
  - Displays the username and a **Logout** button.
- On logout:
  - `AuthService.signout()` is called.
  - User is navigated to `/`.
  - A full page reload is triggered to reset application state.

Signup Flow

- **Component**: `components/signup/signup.ts`
- On successful signup:
  - Authentication errors are cleared.
  - User is redirected to `/signin`.

Signin Flow

- **Component**: `components/signin/signin.ts`
- On successful signin:
  - User is redirected to `/`.
- On authentication failure:
  - Says username and password is wrong

Home → Flight Search

- **Component**: `components/home/home.ts`
- Submits `origin` and `destination` to: FlightService.getFlightByOriginAndDestination()
- Results are rendered using the reusable `FlightList` component.

Booking from Flight List

- **Component**: `components/flight-list/flight-list.ts`

Flow:

1. If the user is not authenticated → redirect to `/signin`.
2. If authenticated:
    - Fetch passenger details using the logged-in user's email.
    - If passenger exists → navigate to `/book` with:
      history.state = { flightId }
    - If passenger does not exist (404) → redirect to `/register`.

Validation Logic

Custom Validators

`usernameValidator`

- **Source**: `src/app/validatorFunctions/usernameValidator.ts`
- Rules:
    - Must start with a letter → `mustStartWithLetter`
    - No spaces allowed → `noSpaces`
    - Allowed characters: `^[A-Za-z][\w]*$` → `invalidChars`
    - Minimum length: 5 → `minLength`

`passwordValidator`

- **Source**: `src/app/validatorFunctions/passwordValidator.ts`
- Rules:
    - Minimum length: 6 → `minLength`
    - No spaces allowed → `noSpaces`

Form-Level Validation

Signup Form

- **Component**: `components/signup/signup.ts`
- Fields:

    - `email`: `required`, `email`
    - `username`: `required`, `usernameValidator`
    - `password`: `required`, `passwordValidator`
- Role handling:
    - Selected roles are stored as enums (e.g., `ROLE_ADMIN`).
    - Before submission, roles are transformed in `AuthService.signup()` by:
        - Removing the `ROLE_` prefix
        - Converting to lowercase (e.g., `admin`)

Signin Form

- **Component**: `components/signin/signin.ts`
- Fields:
    - `username`: `required`, `usernameValidator`
    - `password`: `required`, `passwordValidator`

Home Flight Search Form

- **Component**: `components/home/home.ts`
- Fields:
    - `origin`: `required`, `pattern('^[A-Za-z ]+$')`
    - `destination`: `required`, `pattern('^[A-Za-z ]+$')`

AuthService Integration

- **Source**: `src/app/services/Authentication/auth-service.ts`

Core Responsibilities

- **`currentUser`**:
    - BehaviorSubject-backed observable.
    - Initialized on service construction by calling `/api/auth/me`.
- **`signin()`**:
    - Updates `currentUser` with `UserResponse` containing:
        - `id`, `username`, `email`, `roles`
- **`signout()`**:
    - Clears the `currentUser` BehaviorSubject.
- **`signup()`**:
    - Maps frontend role enums to backend-compatible role strings.

- `error$`:
    - Emits normalized authentication errors.
    - Consumed by the Signup component for user feedback.

Backend code changes:

GET /api/auth/me end point was added in the backend to know the current user logged in.

Frontend uses it to:

- restore login after refresh
- show logout button
- hide signin/signup

`BehaviorSubject<UserResponse | null>` is built around this.

CORS configuration to allow Angular (4200)

Because:

- Angular runs on `http://localhost:4200`
- Backend runs on `http://localhost:8765`
- Browser blocks this by default

So we **allowed it explicitly**.

config.setAllowedOrigins(List.of("http://localhost:4200"));