# Mô tả

Ứng dụng Quản lý sức khỏe cá nhân

* **Mô tả**:  
  Người dùng ghi nhận thông tin dinh dưỡng, bài tập thể thao, giấc ngủ,... App sẽ phân tích và đề xuất cải thiện.  
  More:
  + Chat bot (dùng API AIchat), machine learning khai thác đưa ra các lựa chọn hợp lý (có thể là khai thác tổng hợp kết quả gởi chat bot -> chat bot đưa ra gợi ý)

-> hệ thống tự vấn luôn

* + Thêm role bác sĩ, y tá để biến áp thành áp theo dõi sức khỏe
  + Thêm Quản lý h giấc, lên lịch, gửi thông báo khi đến h,...
* **Công nghệ áp dụng**:  
  + **.NET C#**: API lưu trữ và phân tích dữ liệu sức khỏe.
  + **ETL**: Thu thập thêm dữ liệu từ các nguồn sức khỏe mở (ví dụ API dinh dưỡng, bài tập).
  + **SQL DB**: Lưu hồ sơ người dùng, lịch sử chỉ số cơ thể.
  + **NoSQL DB**: Lưu dữ liệu nhật ký luyện tập hàng ngày (MongoDB - mỗi ngày có thể là 1 document).
  + **React.js**: Giao diện dashboard, báo cáo sức khỏe, biểu đồ lịch sử.

# Thiết kế csdl

Loại thông tin

* Thông tin cá nhân
* **User**
  + Họ tên
  + giới tính
  + ngày sinh
  + Chiều cao
  + cân nặng
  + Thông tin mục tiêu (cân nặng bao nhiêu, (giữ dáng, tăng cơ, giảm cân,....))
* Thông tin sức khỏe (chiều cao cân nặng, BMI, tiền xử bệnh lý, mức độ hoạt động)
  + chỉ số BMI
  + Tiền sử bệnh lý (nếu có)
  + Mức độ hoạt động thể chất trung bình (ít vận động, trung bình, cao)
* Thông tin giấc ngủ
  + Thời gian ngủ
  + Thời gian lên giường
  + Thời gian ngủ dậy
  + Ngày tháng năm
  + Deep sleep
  + REM sleep
  + LIGHT sleep
  + Sleep sound
* Thông tin tập luyện
  + Thời gian tập
  + Bài tập
  + Calo tiêu thụ
* Thông tin dinh dưỡng hằng ngày
  + Bữa ăn: sáng / trưa / tối / ăn nhẹ
  + Thực phẩm đã dùng (tên món, khẩu phần, đơn vị, thời gian)
  + Giá trị dinh dưỡng: calo, protein, carbs, chất béo, chất xơ...
  + Dữ liệu có thể được bổ sung từ API dinh dưỡng như Edamame, USDA...
* **Postgresql**
* Authorization

CREATE TABLE AspNetRoles (

Id **uuid** PRIMARY KEY,

Name **VARCHAR**(256) NOT NULL,

NormalizedName **VARCHAR**(256) NOT NULL,

ConcurrencyStamp **VARCHAR**(256),

CreatedAt **TIMESTAMP** DEFAULT CURRENT\_TIMESTAMP,

UpdatedAt **TIMESTAMP** DEFAULT CURRENT\_TIMESTAMP

);

CREATE TABLE AspNetUsers (

Id **uuid** PRIMARY KEY,

UserName **VARCHAR**(256) NOT NULL,

NormalizedUserName **VARCHAR**(256) NOT NULL,

Email **VARCHAR**(256) NOT NULL,

NormalizedEmail **VARCHAR**(256) NOT NULL,

EmailConfirmed **BOOLEAN** DEFAULT FALSE,

PasswordHash **VARCHAR**(256),

SecurityStamp **VARCHAR**(256),

ConcurrencyStamp **VARCHAR**(256),

PhoneNumber **VARCHAR**(256),

PhoneNumberConfirmed **BOOLEAN** DEFAULT FALSE,

TwoFactorEnabled **BOOLEAN** DEFAULT FALSE,

LockoutEnd **TIMESTAMP**,

LockoutEnabled **BOOLEAN** DEFAULT TRUE,

AccessFailedCount **INT** DEFAULT 0,

CreatedAt **TIMESTAMP** DEFAULT CURRENT\_TIMESTAMP,

UpdatedAt **TIMESTAMP** DEFAULT CURRENT\_TIMESTAMP

);

* **MongoDB**
* **Thông tin cá nhân**

{

"\_id": ObjectId("..."),

"user\_id": 123, // Liên kết với PostgreSQL

"profile": {

"full\_name": "Nguyễn Văn A",

"gender": "Male",

"dob": "1995-10-10",

"height": 175,

"weight": 70,

"bmi": 22.9,

"goals": "Tăng cơ"

},

* **Sleep logs**

{

"user\_id": 1,

"date": "2025-05-19",

"bed\_time": "22:30",

"wake\_time": "06:45",

"deep\_sleep": 120,

"rem\_sleep": 90,

"light\_sleep": 150,

"sleep\_score": 80

}

* **Workouts**

{

"user\_id": 1,

"date": "2025-05-19",

"workouts": [

{

"name": "Running",

"duration\_minutes": 30,

"calories\_burned": 250

},

{

"name": "Push-ups",

"duration\_minutes": 10,

"calories\_burned": 50

}

]

}

* **Daily meals**

{

"user\_id": 1,

"date": "2025-05-19",

"meals": [

{

"type": "breakfast",

"time": "07:30",

"foods": [

{

"name": "Oatmeal",

"portion": 1,

"unit": "bowl",

"nutrition": {

"calories": 150,

"protein": 5,

"carbs": 27,

"fat": 3,

"fiber": 4

}

}

]

},

{

"type": "dinner",

"time": "18:30",

"foods": [

{

"name": "Grilled chicken",

"portion": 150,

"unit": "g",

"nutrition": {

"calories": 300,

"protein": 30,

"carbs": 0,

"fat": 15,

"fiber": 0

}

}

]

}

]

}

* Nutritious sources

{

"name": "Avocado",

"unit": "100g",

"nutrition": {

"calories": 160,

"protein": 2,

"carbs": 9,

"fat": 15,

"fiber": 7

}

}

* **Redis**
* Session (cache)
* Key: session:user:{user\_id}
* Value: access token, role, expiry, last active

{

"access\_token": "eyJhbGciOiJIUzI1NiIsInR...",

"role": "user",

"expiry": "2025-05-21T15:00:00Z",

"last\_active": "2025-05-20T10:22:35Z"

}

* Tổng kết sức khỏe theo ngày
* Key: analysis:user:{user\_id}:summary:2025-05-19
* Value: quick JSON object:

{

"calories\_in": 1800,

"calories\_out": 350,

"sleep\_score": 82,

"suggestions": ["Tăng thời gian chạy", "Giảm carbs bữa tối"]

}

1. **Cài đặt**

## Docker (migration\_script)

Đây là ý tưởng sử dụng file batch để chạy docker-compose up với các database đã được cầu hình sẵn và sau đó sẽ đọc các file dữ liệu tương ứng với các loại database tương ứng. Mục đích ban đầu là để quản lý các phiên bản dữ liệu của nhiều developer cùng phát triển, nhưng trong đồ án cá nhân này chỉ để thuận tiện cho việc quan lý và khởi tạo các csdl tương ứng

### **Postgres**

### Redis

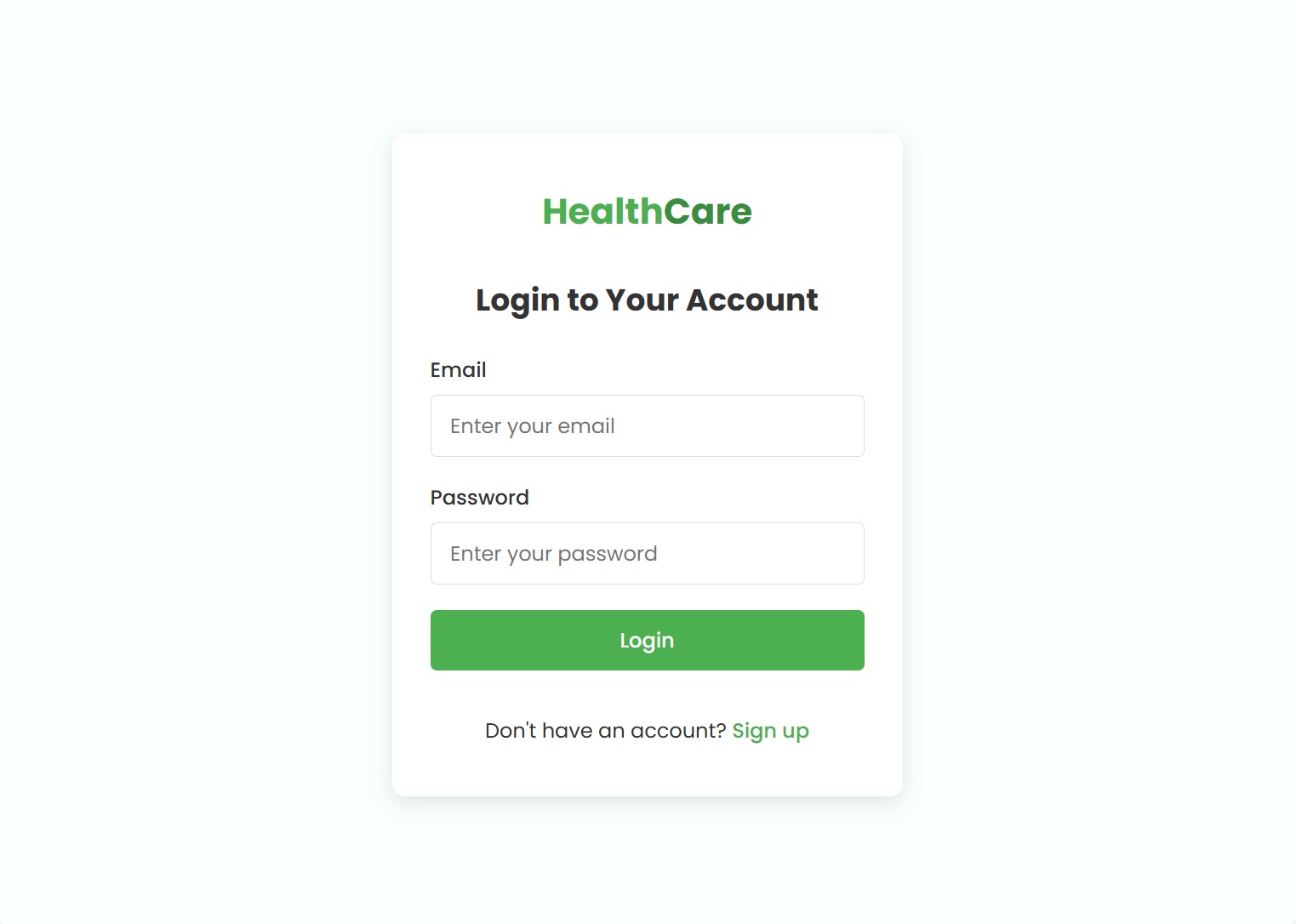
### **MongoDB**

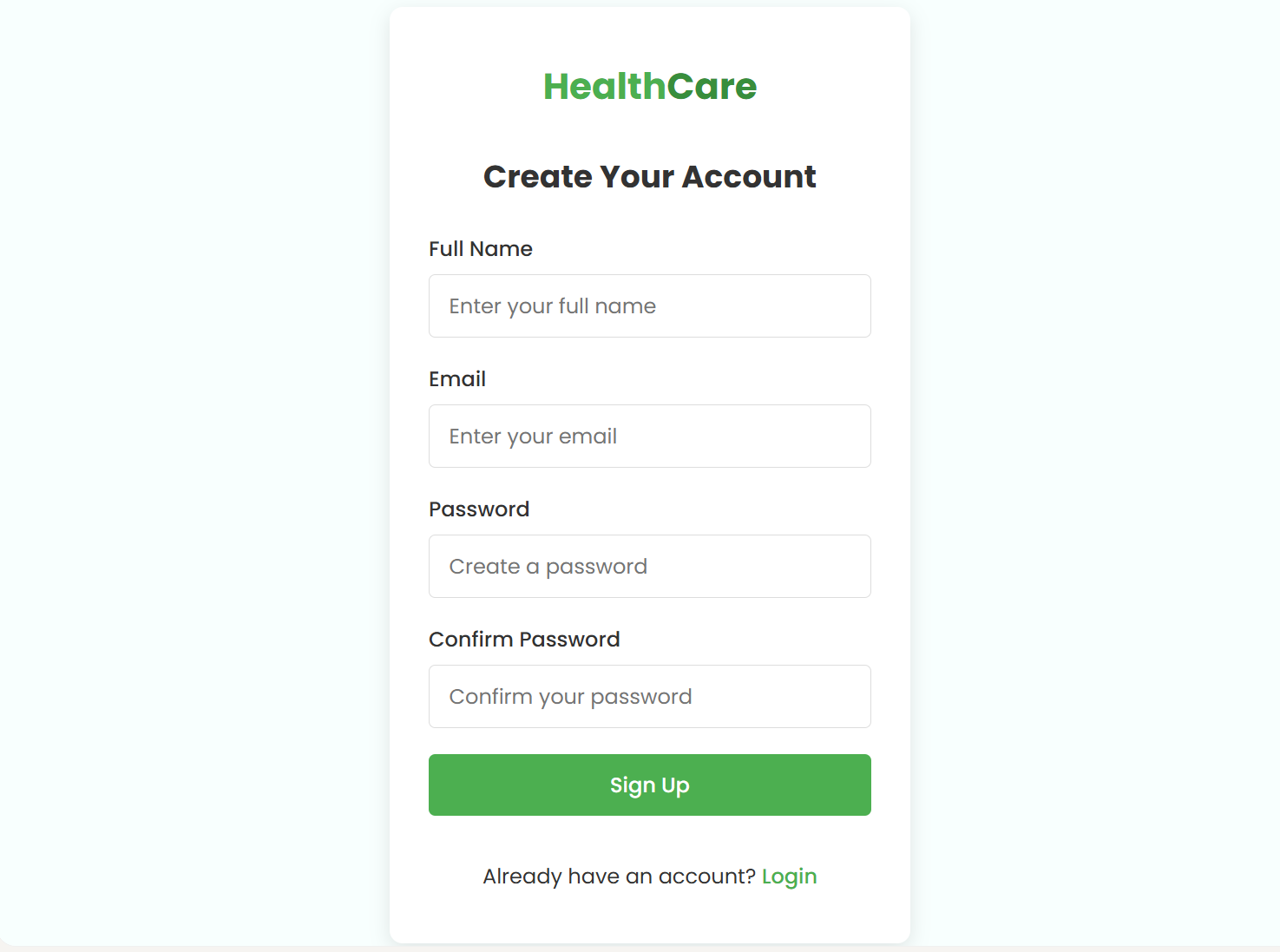
## **Azure**

1. **Authorization & Authentication**

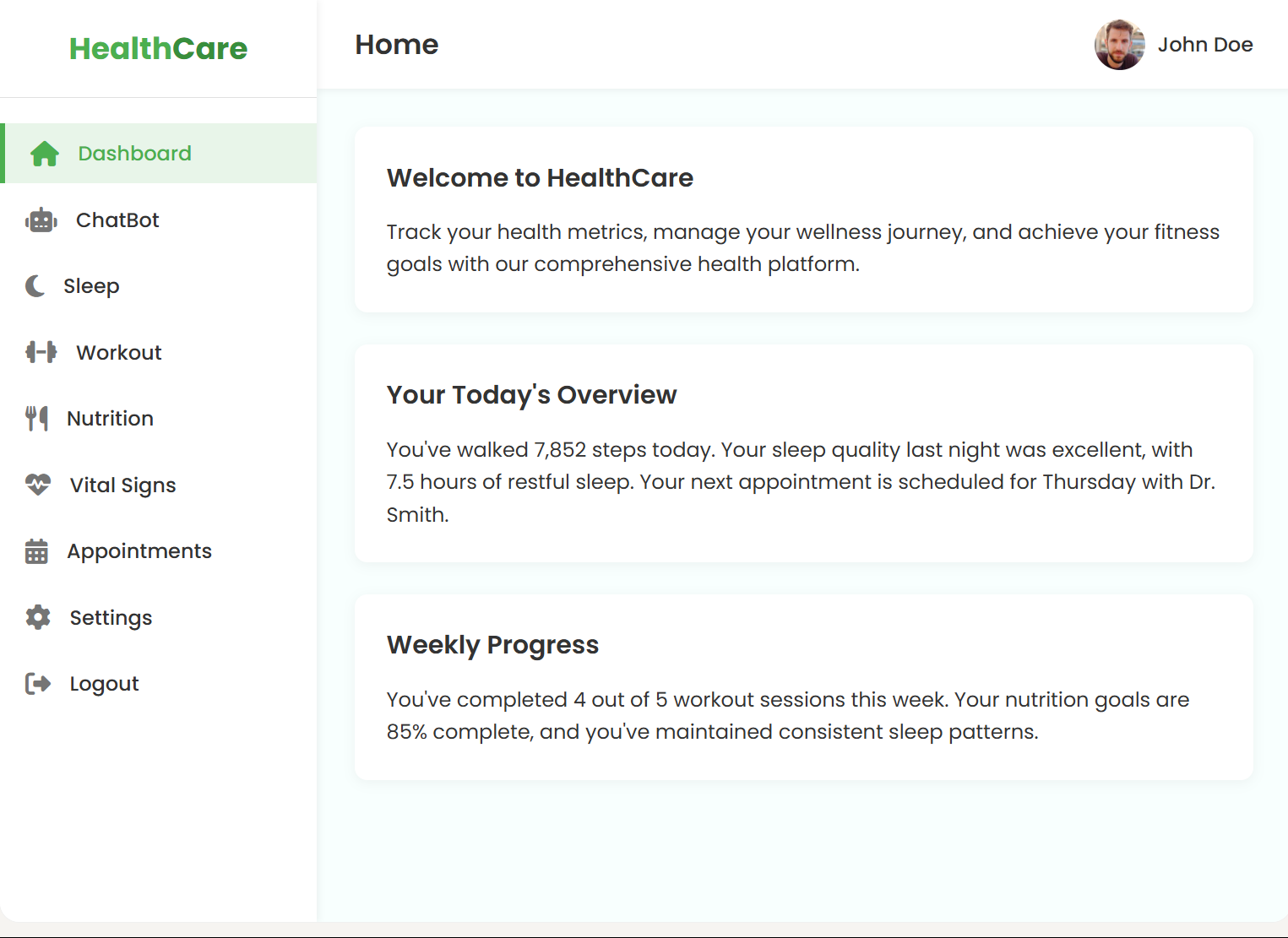
Sử dụng JWT(Json web token) để phát sinh token xác thực người dùng đồng thời phân cũng có vai trò phân quyền khi đăng nhập.

1. **Thiết kế giao diện**
2. Đăng ký và đăng nhập

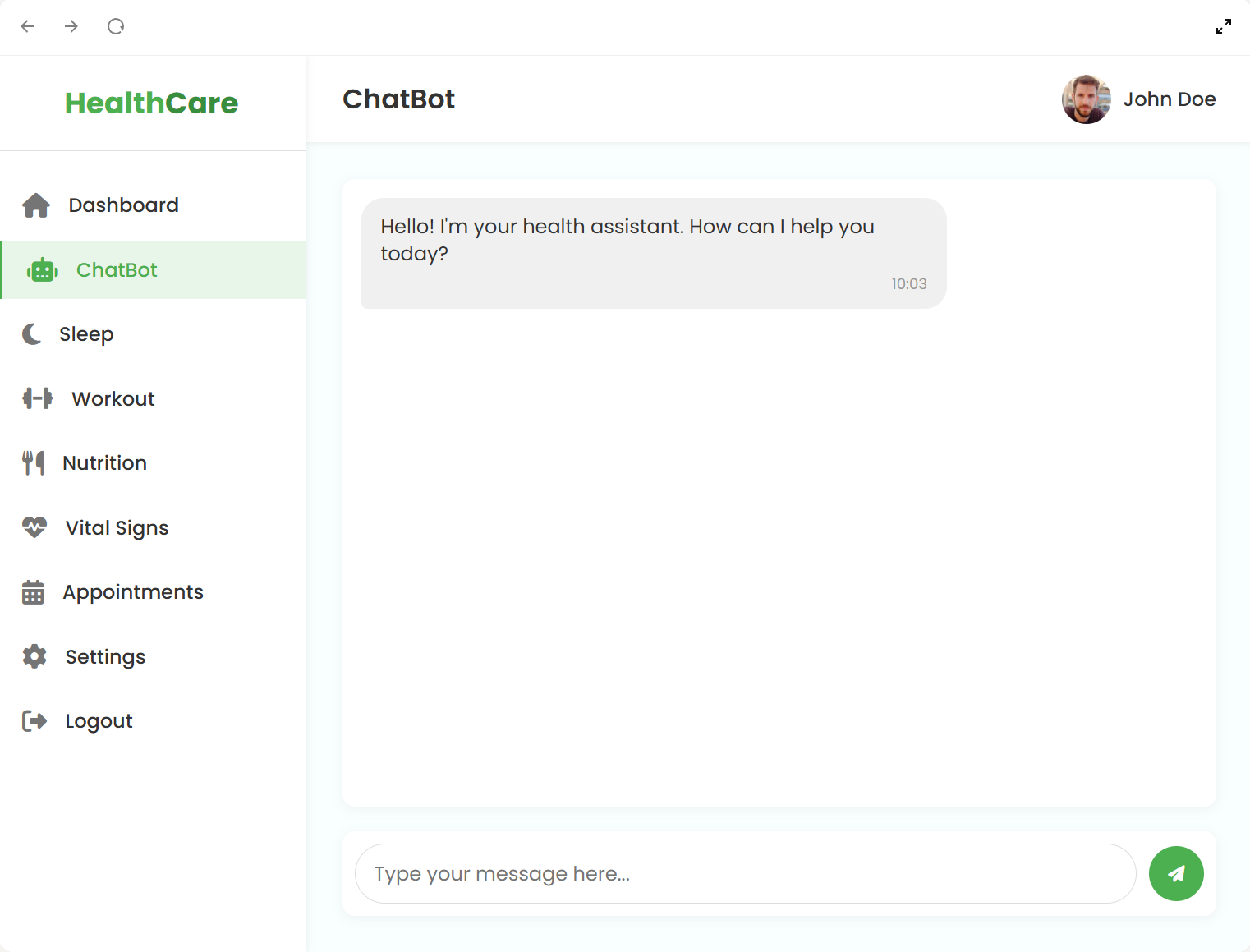




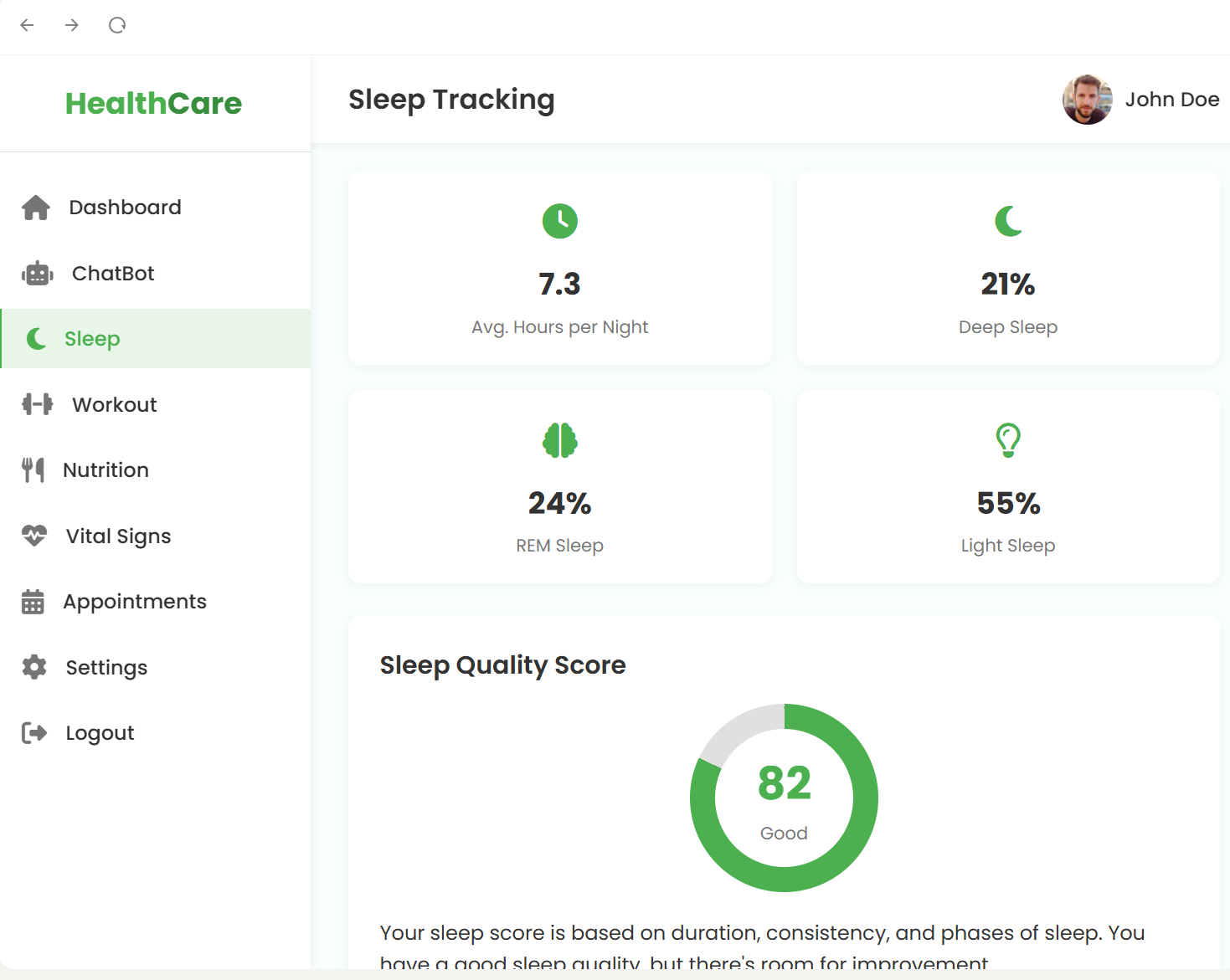
1. Home page

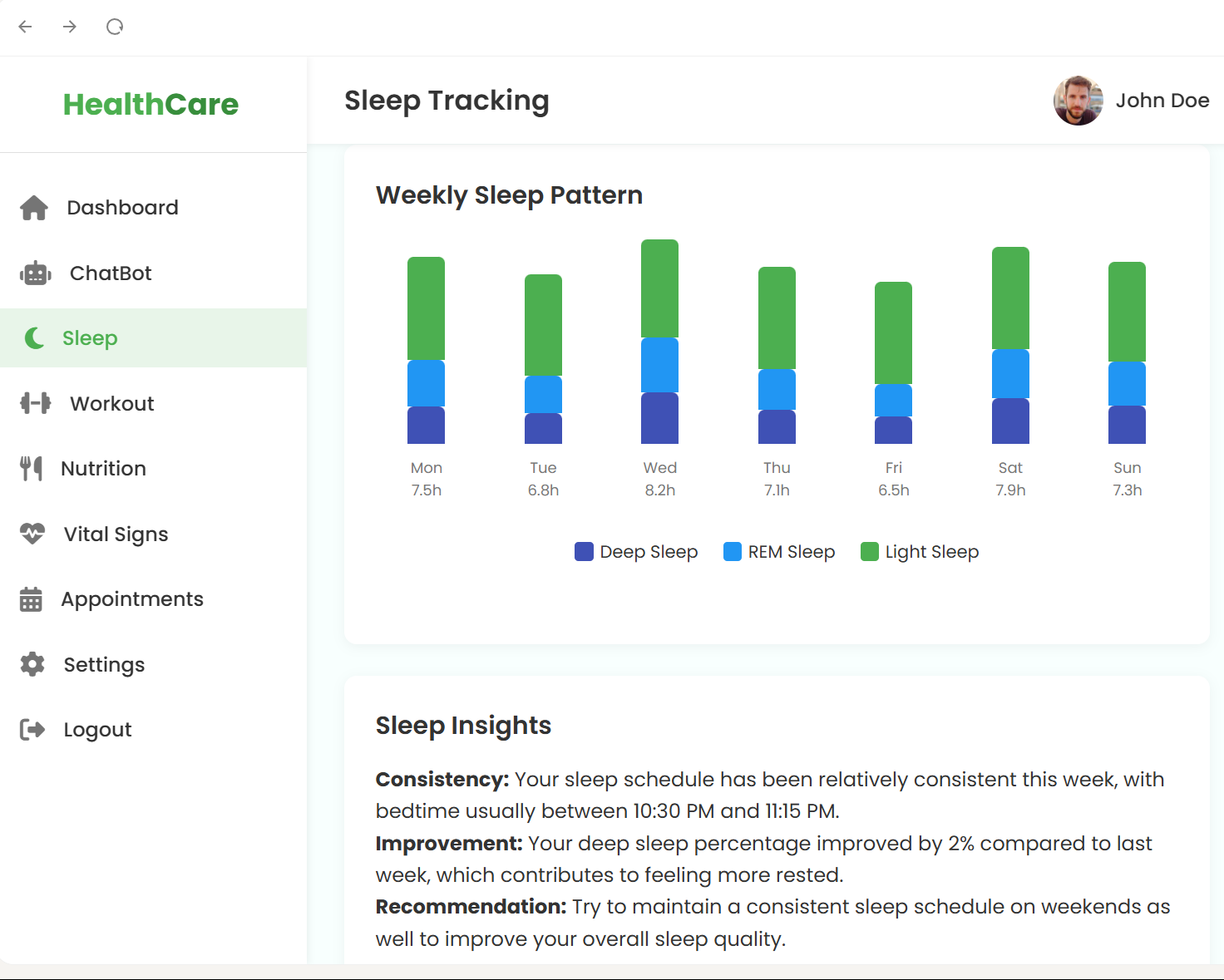


1. Chatbot

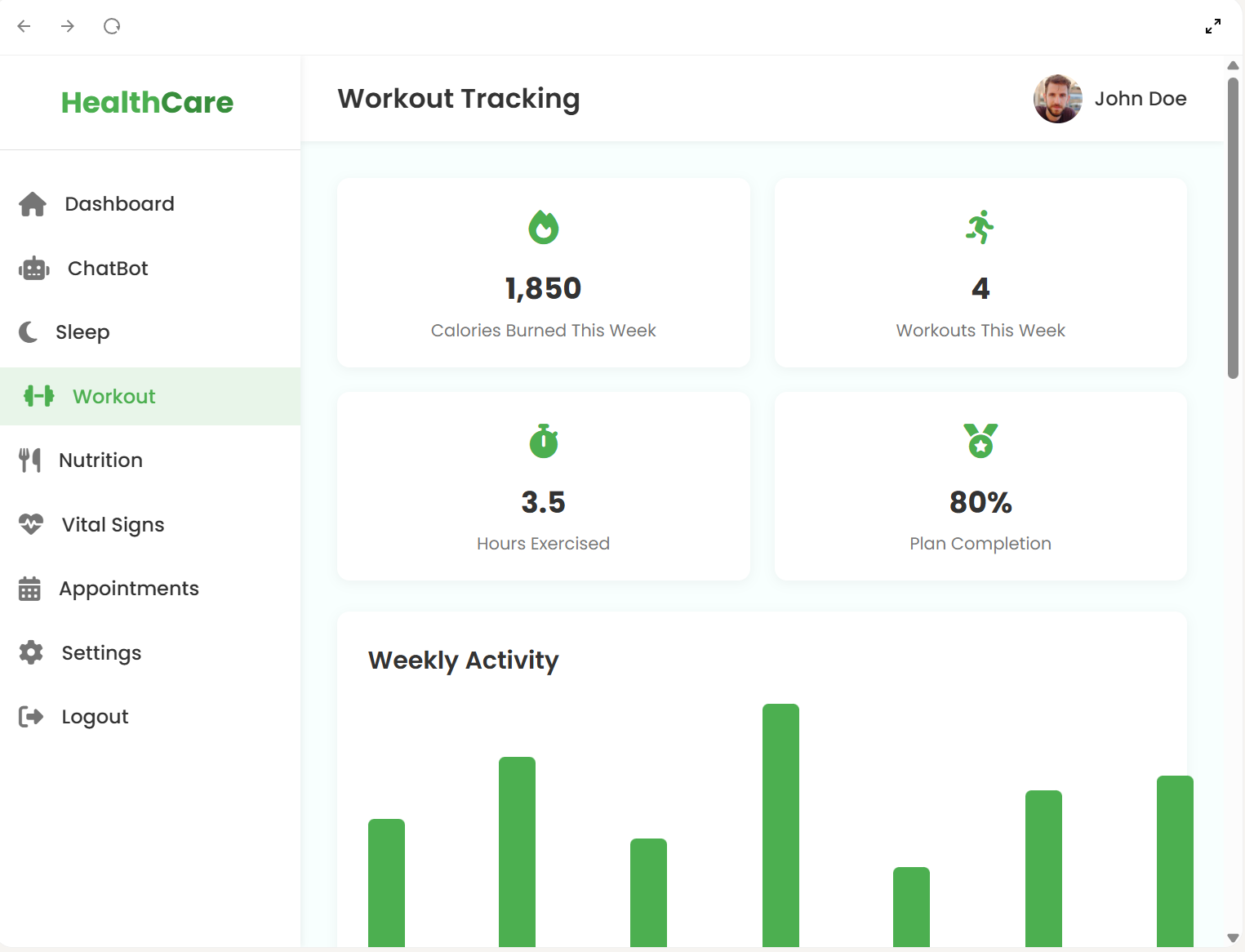


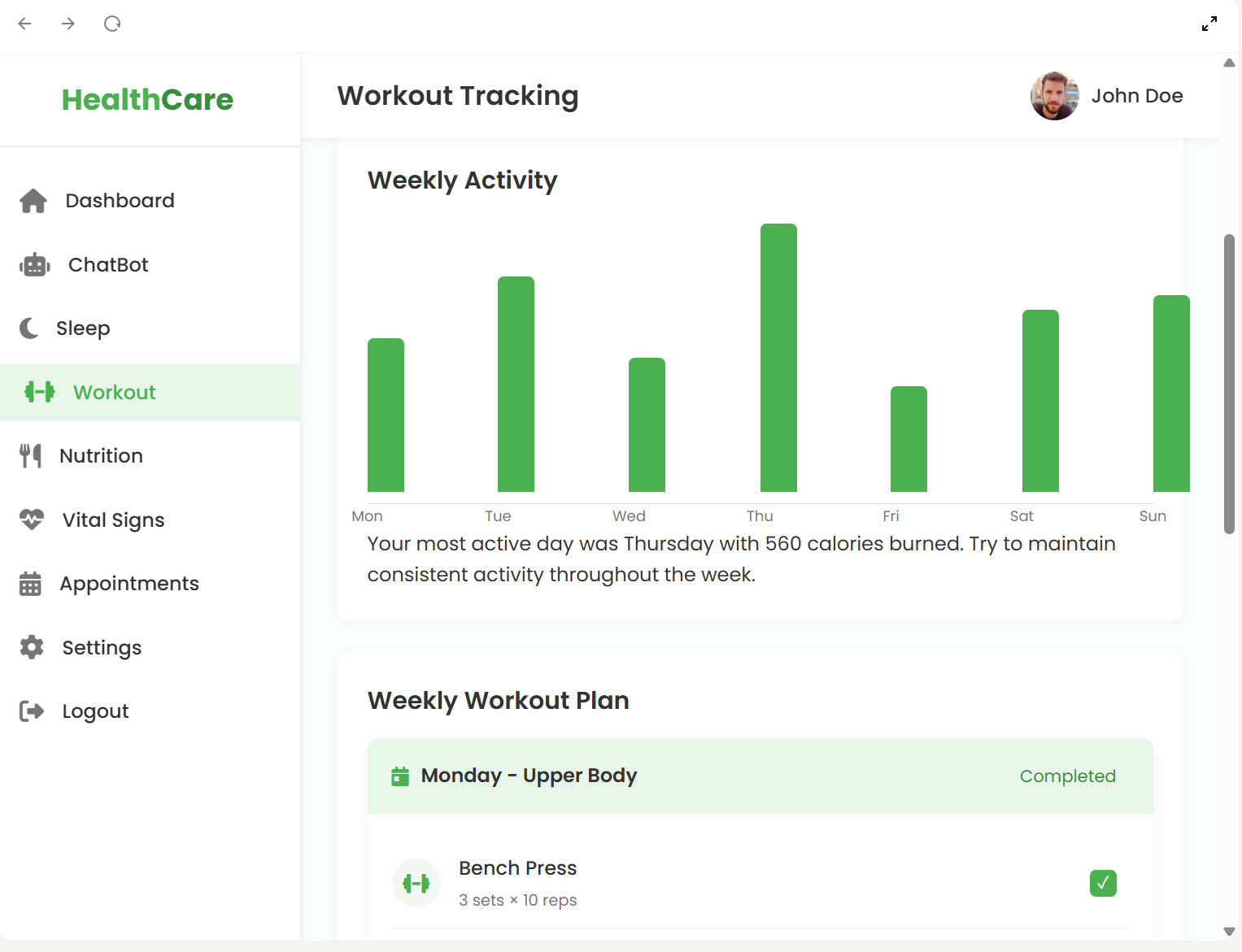
1. Sleep

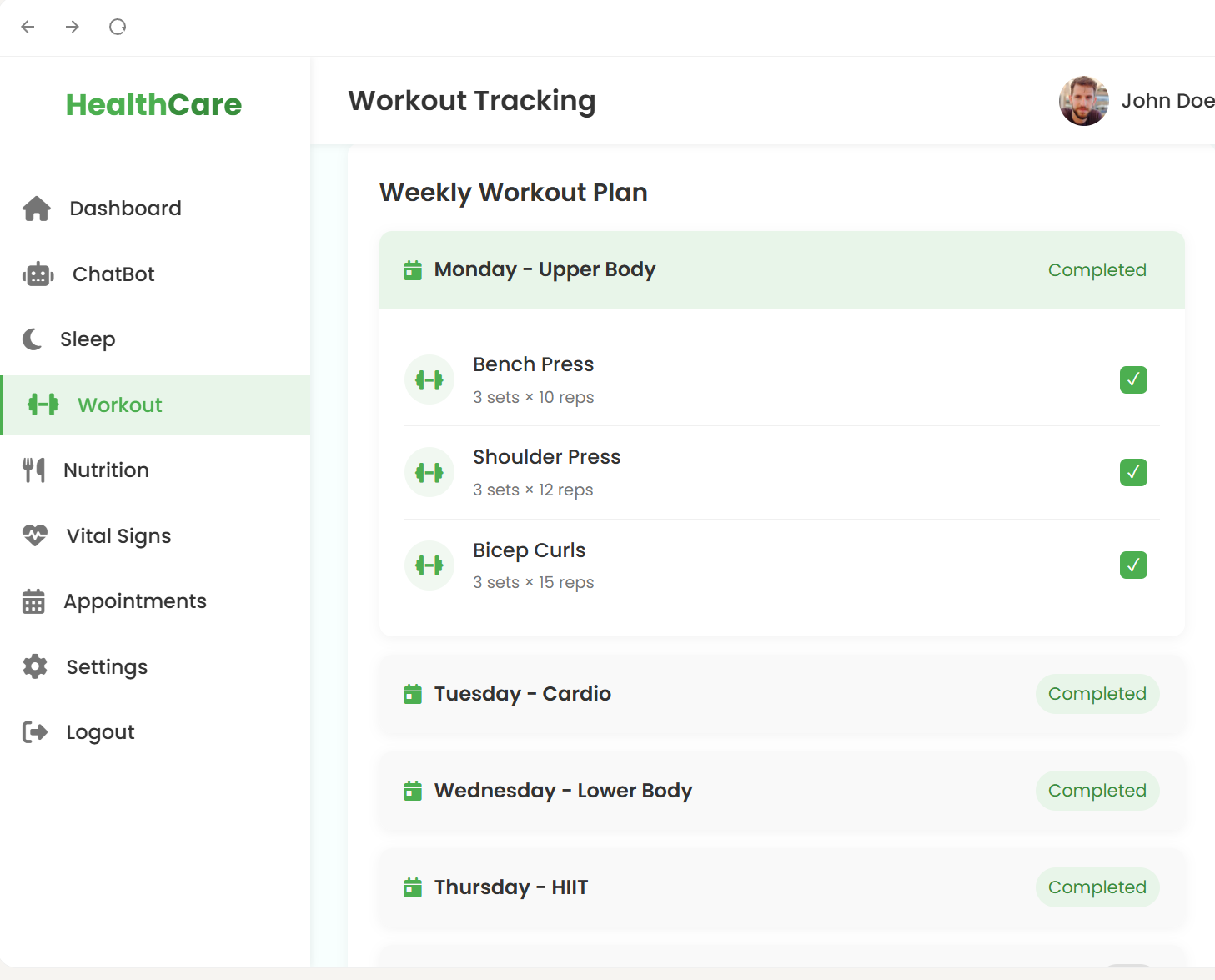


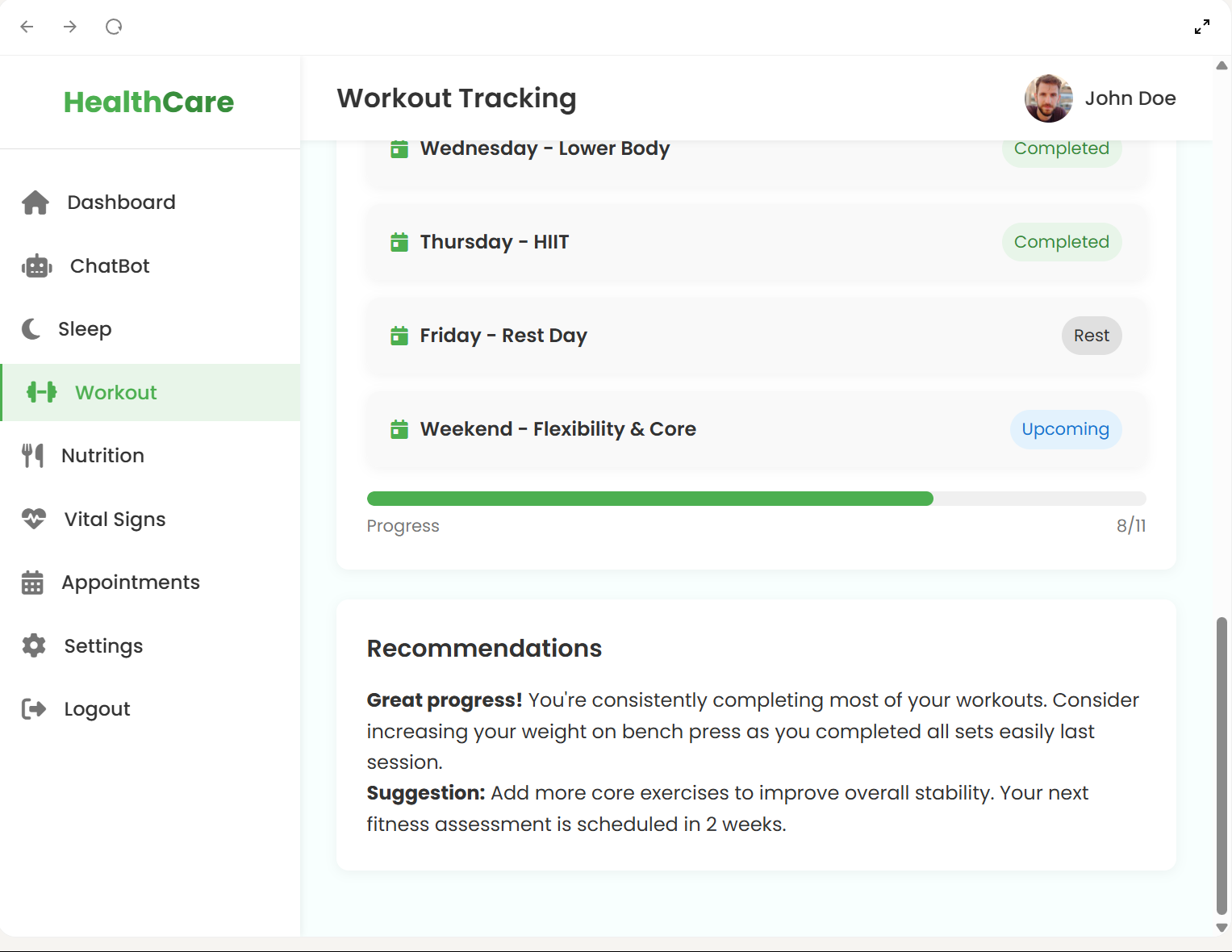


1. Workout

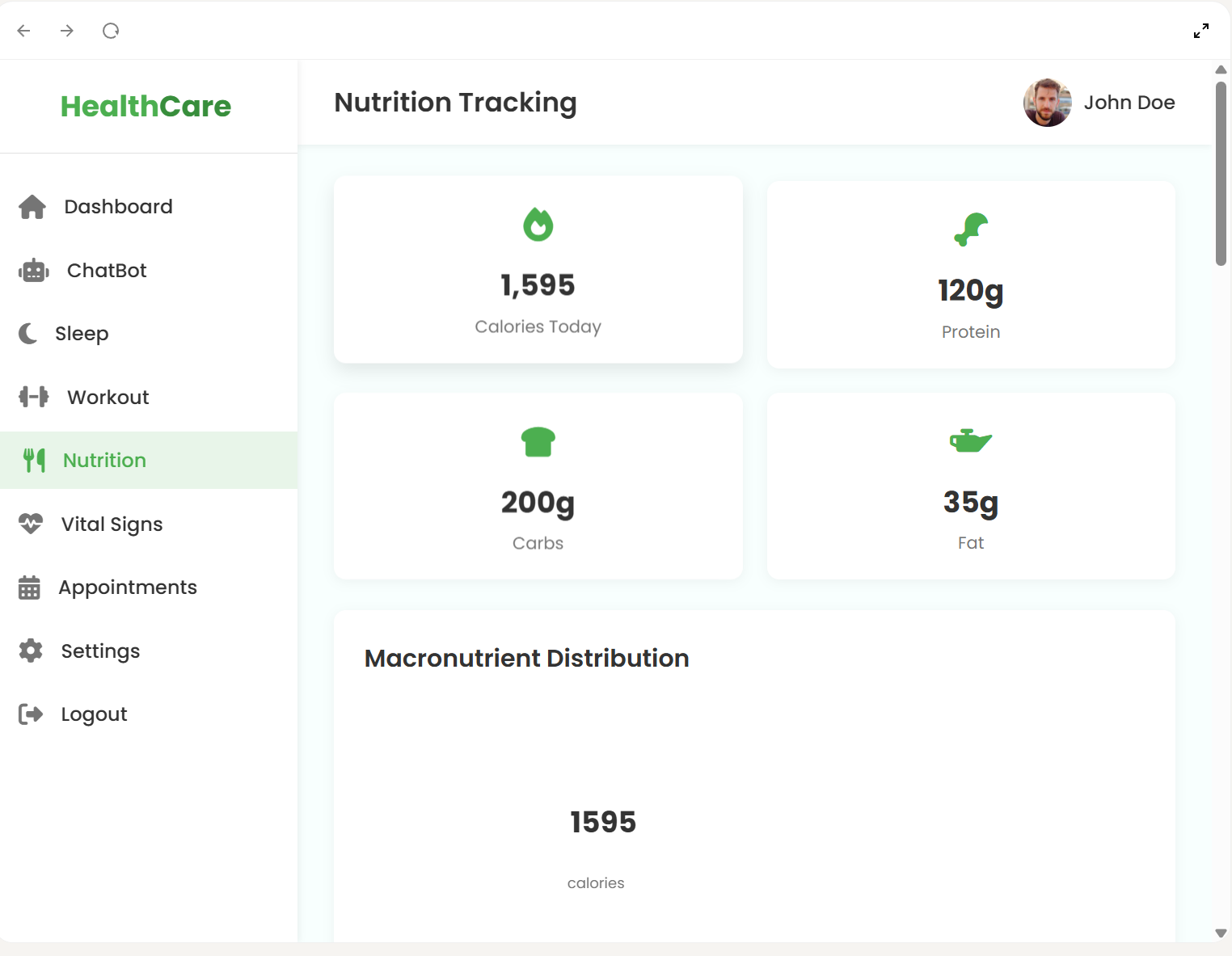


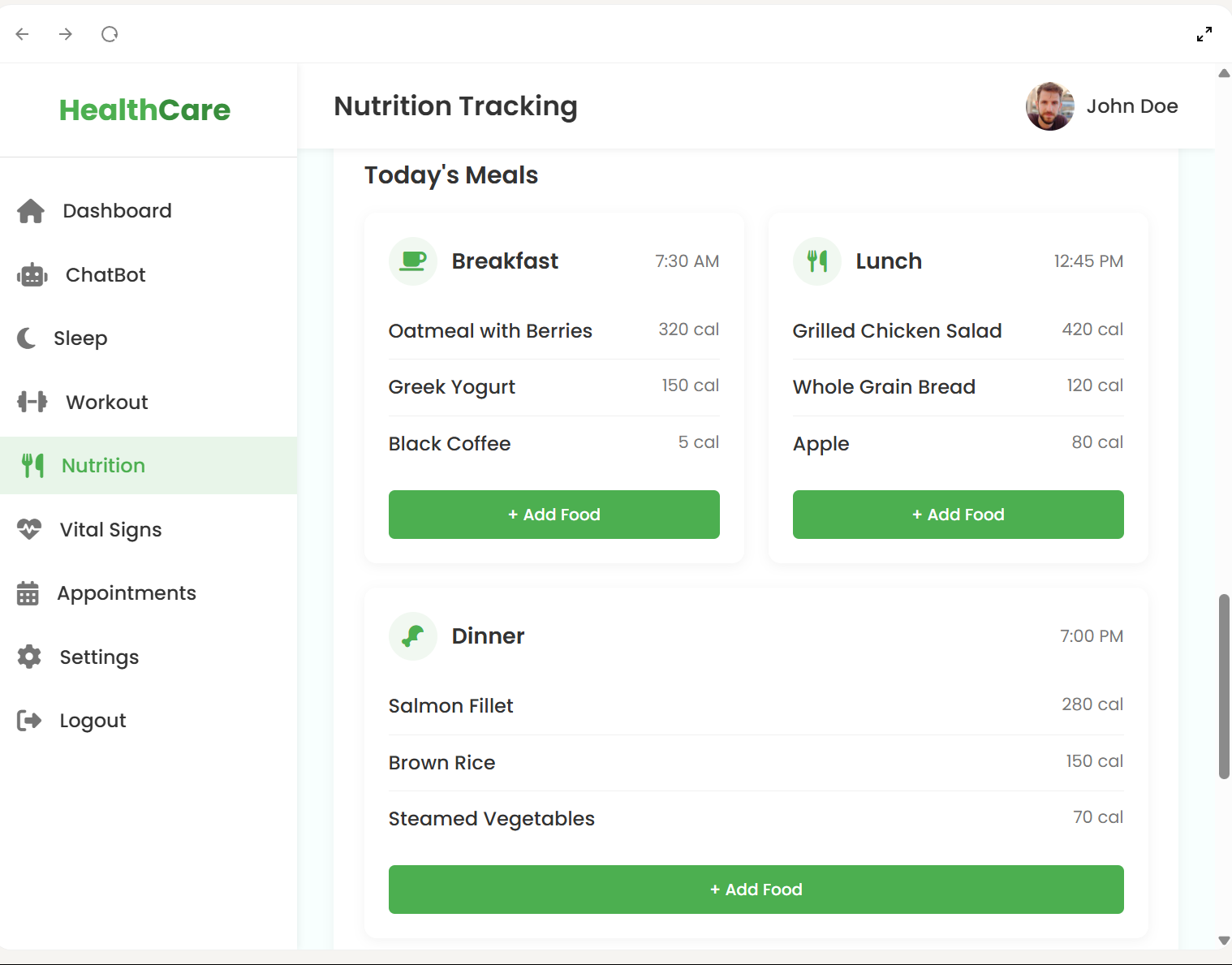


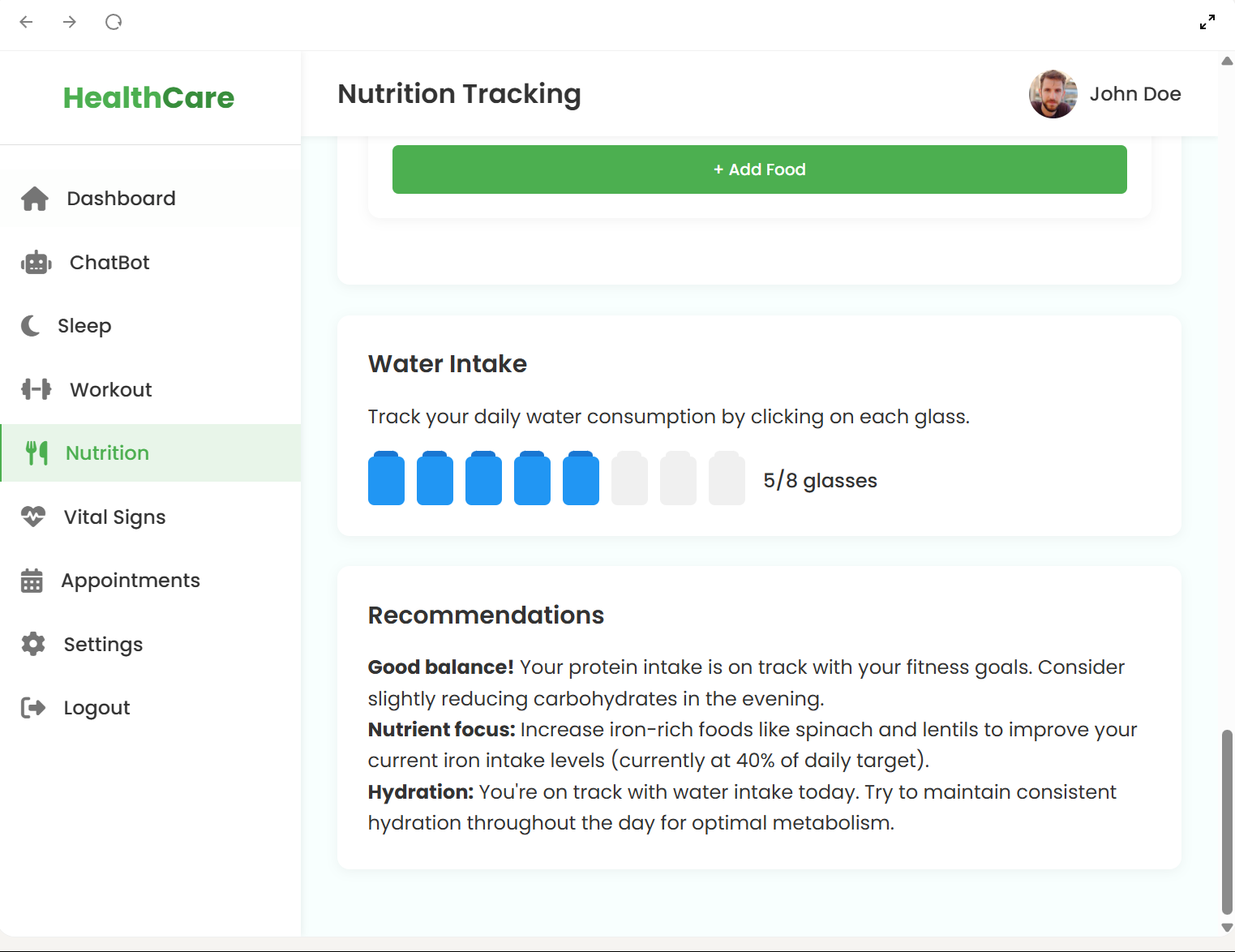




1. Nutrition







1. **Kết quả**