**Smart Calendar – Cloud Computing 2025**

**Ștefan Andrei, Grupa 1132**

**Link-uri Proiect**

* **🎥 Video prezentare**: [Smart Calendar Demo](https://www.youtube.com/watch?v=sVOJogU5xaY)
* **🌐 Link publicare**: [Smart Calendar Live](https://smart-calendar-papi.onrender.com)
* **📁 Repository Git**: [GitHub Repository](https://github.com/anro772/smart-calendar)

**1. Introducere**

Smart Calendar este o aplicație modernă de calendar care folosește tehnologii AI pentru a facilita gestionarea programului personal și profesional. Aplicația oferă o interfață intuitivă și elegantă, fiind dezvoltată cu React pe partea de frontend și Express.js pentru backend, integrând API-ul Gemini de la Google pentru caracteristicile sale inteligente, alături de Firebase pentru autentificare și persistența datelor.

**2. Descriere Problemă**

**Problema identificată**

Utilizatorii moderni se confruntă cu dificultăți în gestionarea eficientă a programului zilnic, având nevoie de:

* **Planificare inteligentă**: Ajutor în organizarea evenimentelor și prioritizarea activităților
* **Sugestii contextuale**: Recomandări personalizate bazate pe tipul și descrierea evenimentelor
* **Sincronizare cross-platform**: Acces la calendar de pe orice dispozitiv
* **Rezumate inteligente**: Privire de ansamblu asupra programului săptămânal/lunar

**Soluția propusă**

Smart Calendar rezolvă aceste probleme prin:

* Integrarea AI pentru sugestii automate și personalizate
* Interfață responsive și intuitivă
* Sincronizare în timp real prin Firebase
* Generare automată de rezumate și priorități

**3. Descriere API**

**Servicii Cloud Utilizate**

**3.1 Google Generative AI (Gemini 2.0)**

* **Scop**: Generare de sugestii inteligente și rezumate pentru evenimente
* **Autentificare**: API Key
* **Endpoint**: Serviciu cloud pentru procesare de limbaj natural

**3.2 Firebase**

* **Servicii utilizate**:
  + **Authentication**: Autentificare cu email/parolă și Google OAuth
  + **Firestore Database**: Persistența datelor pentru evenimente și utilizatori
* **Autentificare**: Firebase Config cu credențiale API

**API-ul Aplicației (Backend Express.js)**

**Endpoint-uri principale:**

**Base URL**: http://localhost:5000/api (development) / https://smart-calendar-backend-30re.onrender.com (production)

**4. Flux de Date**

**4.1 Arhitectura Aplicației**

Frontend (React) ↔ Backend (Express.js) ↔ Google Gemini AI

↕

Firebase (Auth + Firestore)

**4.2 Fluxul de Autentificare**

1. **Frontend**: Utilizatorul se autentifică prin Firebase Auth
2. **Firebase**: Validează credențialele și returnează token
3. **Frontend**: Stochează token-ul și redirecționează către dashboard
4. **Firestore**: Sincronizează datele utilizatorului

**4.3 Fluxul pentru Sugestii AI**

**Request - POST /api/suggestions**

{

"description": "Întâlnire cu echipa de dezvoltare pentru planificarea sprintului"

}

**Response**

{

"suggestions": "Pregătește agenda întâlnirii dinainte. Solicită update-uri de la fiecare membru al echipei. Alocă timp pentru discuții despre blocaje.",

"color": "#8b5cf6"

}

**4.4 Fluxul pentru Rezumat Evenimente**

**Request - POST /api/summary**

{

"events": [

{

"id": "1",

"title": "Întâlnire echipă",

"start": "2025-05-21T09:00:00",

"end": "2025-05-21T10:00:00",

"description": "Planificare sprint"

},

{

"id": "2",

"title": "Prezentare proiect",

"start": "2025-05-22T14:00:00",

"end": "2025-05-22T15:30:00",

"description": "Demo pentru client"

}

]

}

**Response**

{

"summary": "<div class=\"structured-summary\">...</div>",

"rawData": {

"overview": "Ai o săptămână încărcată cu 2 evenimente importante, inclusiv întâlniri de echipă și prezentări.",

"events": [

{

"title": "Întâlnire echipă",

"date": "5/21/2025",

"time": "9:00:00 AM - 10:00:00 AM",

"description": "Planificare sprint",

"key\_points": ["Pregătește agenda și prioritățile pentru sprint"]

}

],

"preparations": [

"Pregătește materialele pentru întâlnirea de echipă",

"Verifică disponibilitatea participanților",

"Creează agenda detaliată pentru prezentare"

],

"priorities": [

"Finalizează pregătirile pentru prezentarea de mâine",

"Coordonează cu echipa pentru alignment",

"Testează demo-ul înainte de prezentare"

]

}

}

**4.5 Metode HTTP Utilizate**

* **GET**: Pentru citirea datelor (implicite în Firebase SDK)
* **POST**: Pentru crearea evenimentelor și solicitarea serviciilor AI
* **PUT**: Pentru actualizarea evenimentelor existente
* **DELETE**: Pentru ștergerea evenimentelor

**4.6 Autentificare și Autorizare**

**Firebase Authentication**

* **Email/Password**: Autentificare tradițională
* **Google OAuth**: Autentificare cu cont Google
* **Token Management**: JWT tokens pentru sesiuni

**API Security**

// Exemplu middleware autentificare

const authenticateUser = async (req, res, next) => {

const token = req.headers.authorization?.split(' ')[1];

try {

const decodedToken = await admin.auth().verifyIdToken(token);

req.user = decodedToken;

next();

} catch (error) {

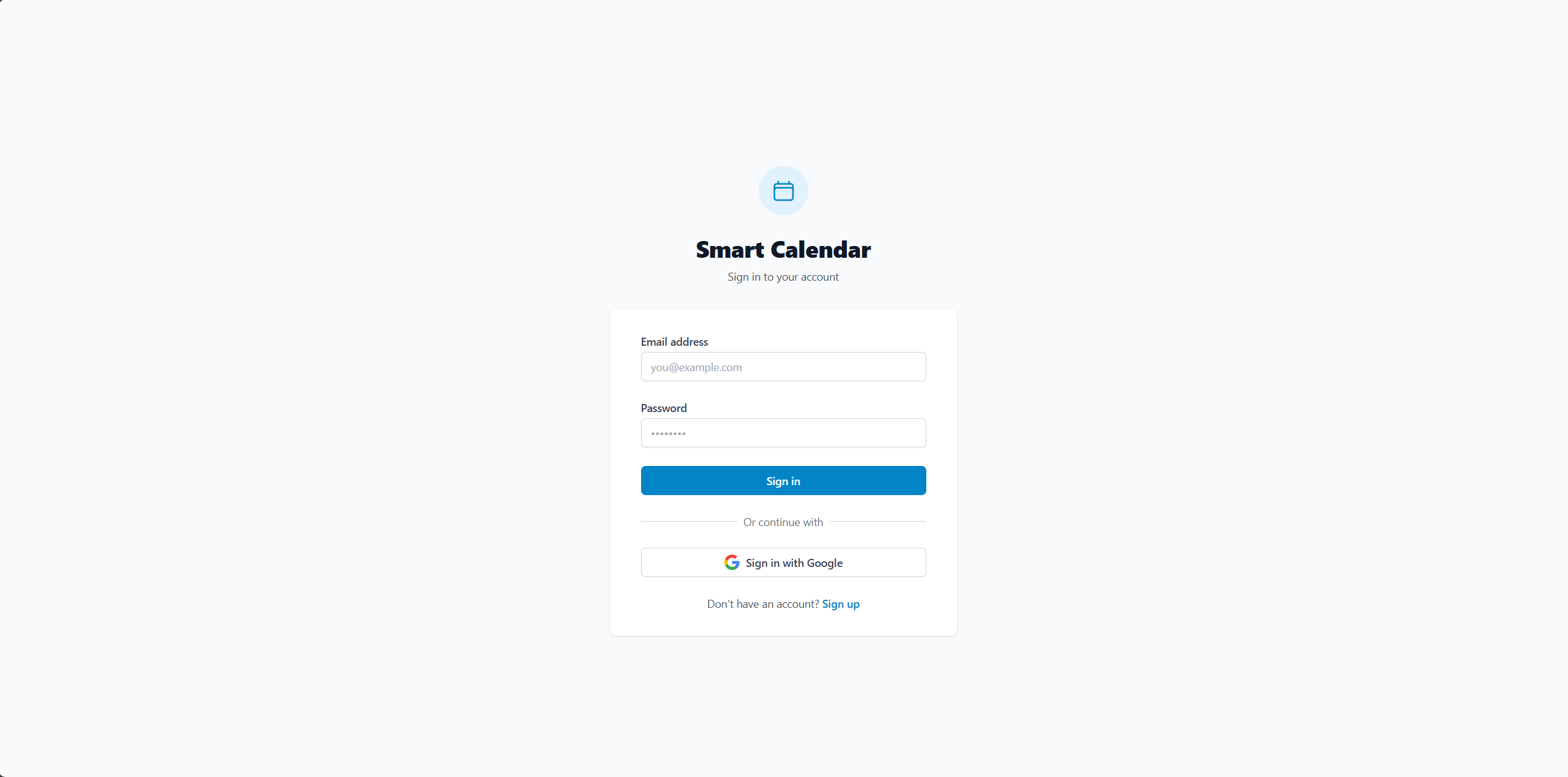
res.status(401).json({ error: 'Unauthorized' });

}

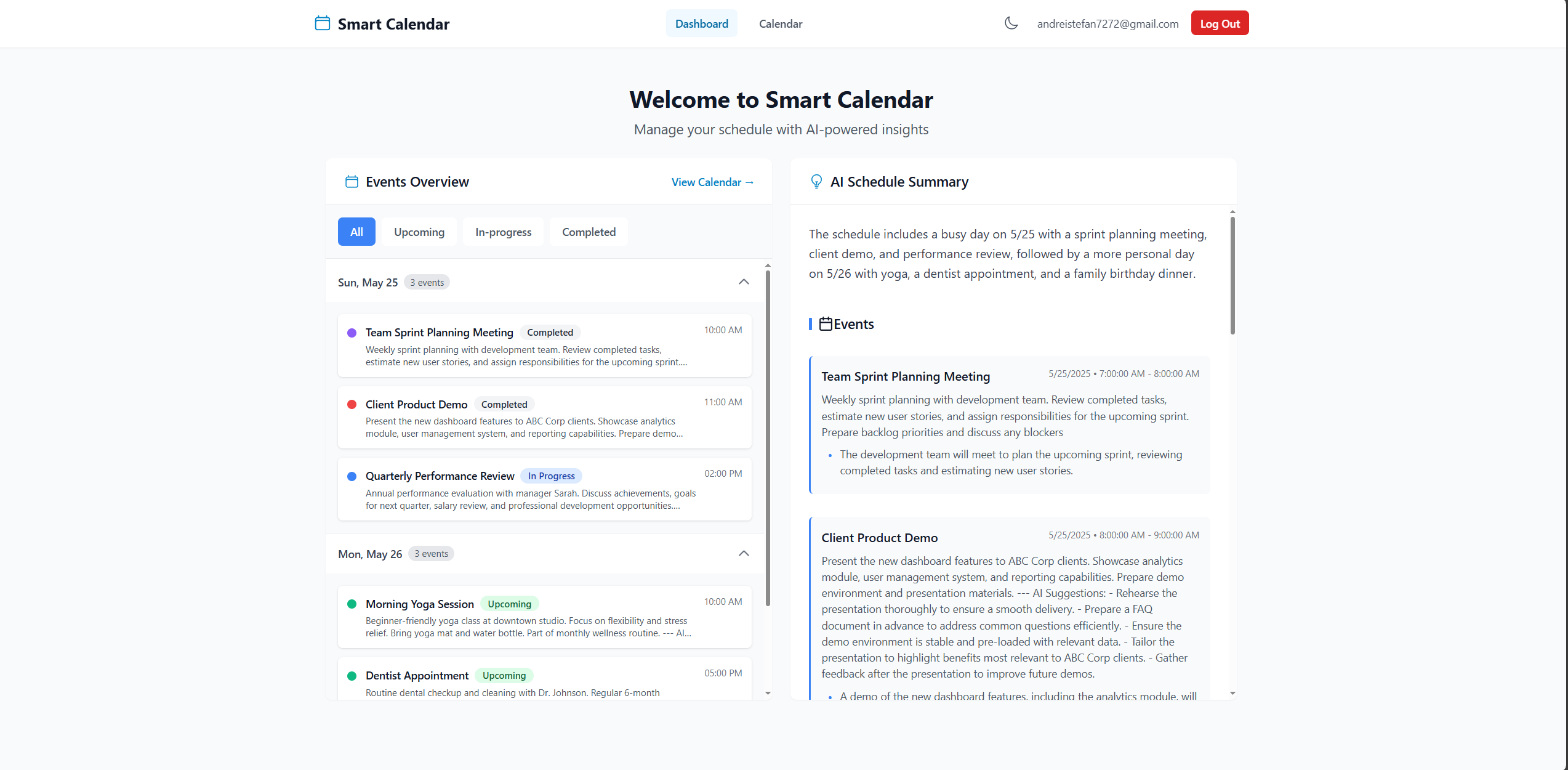
};

**5. Capturi Ecran Aplicație**

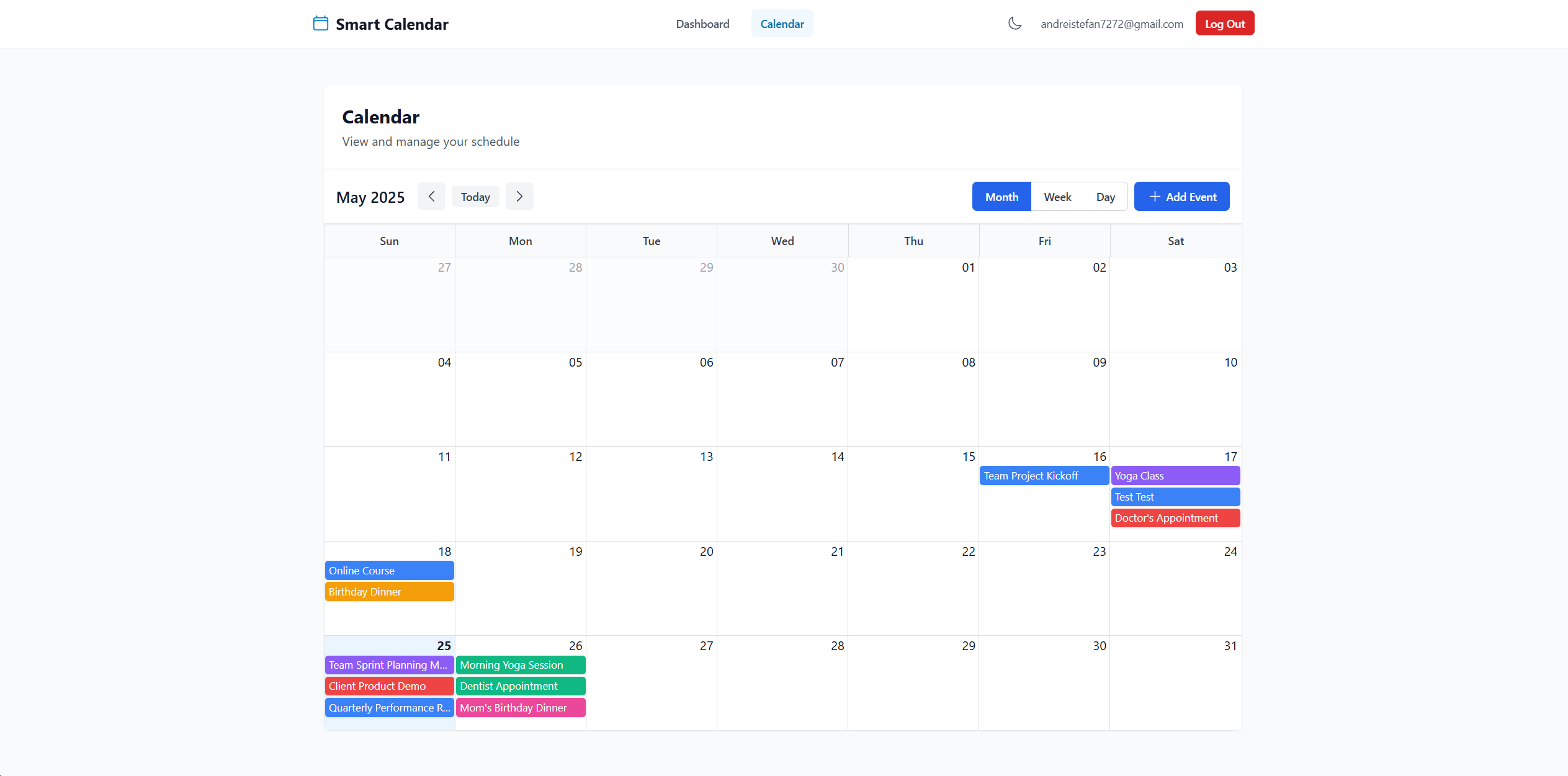
**5.1 Pagina de Autentificare**

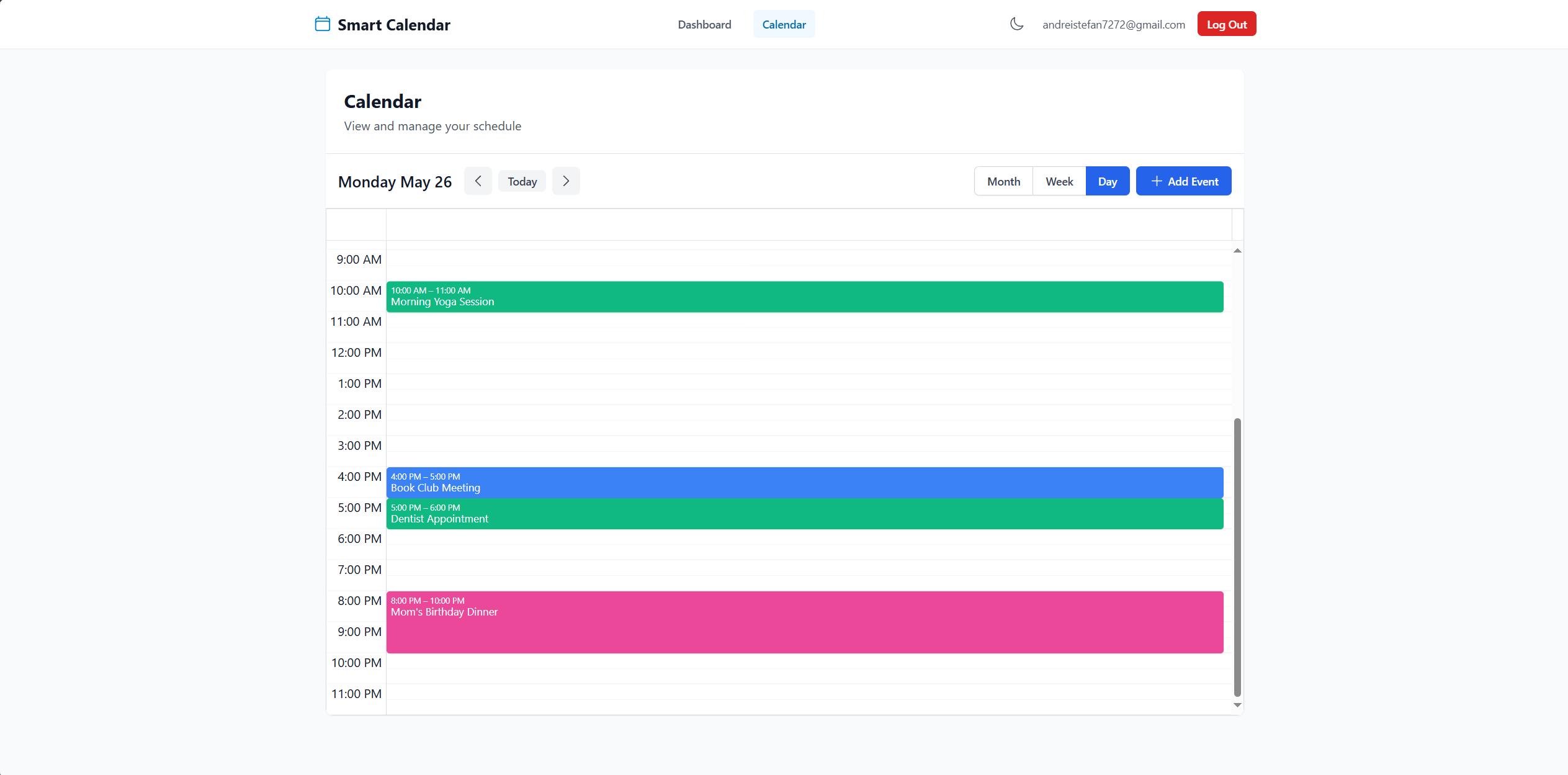
****

**5.2 Dashboard Principal**

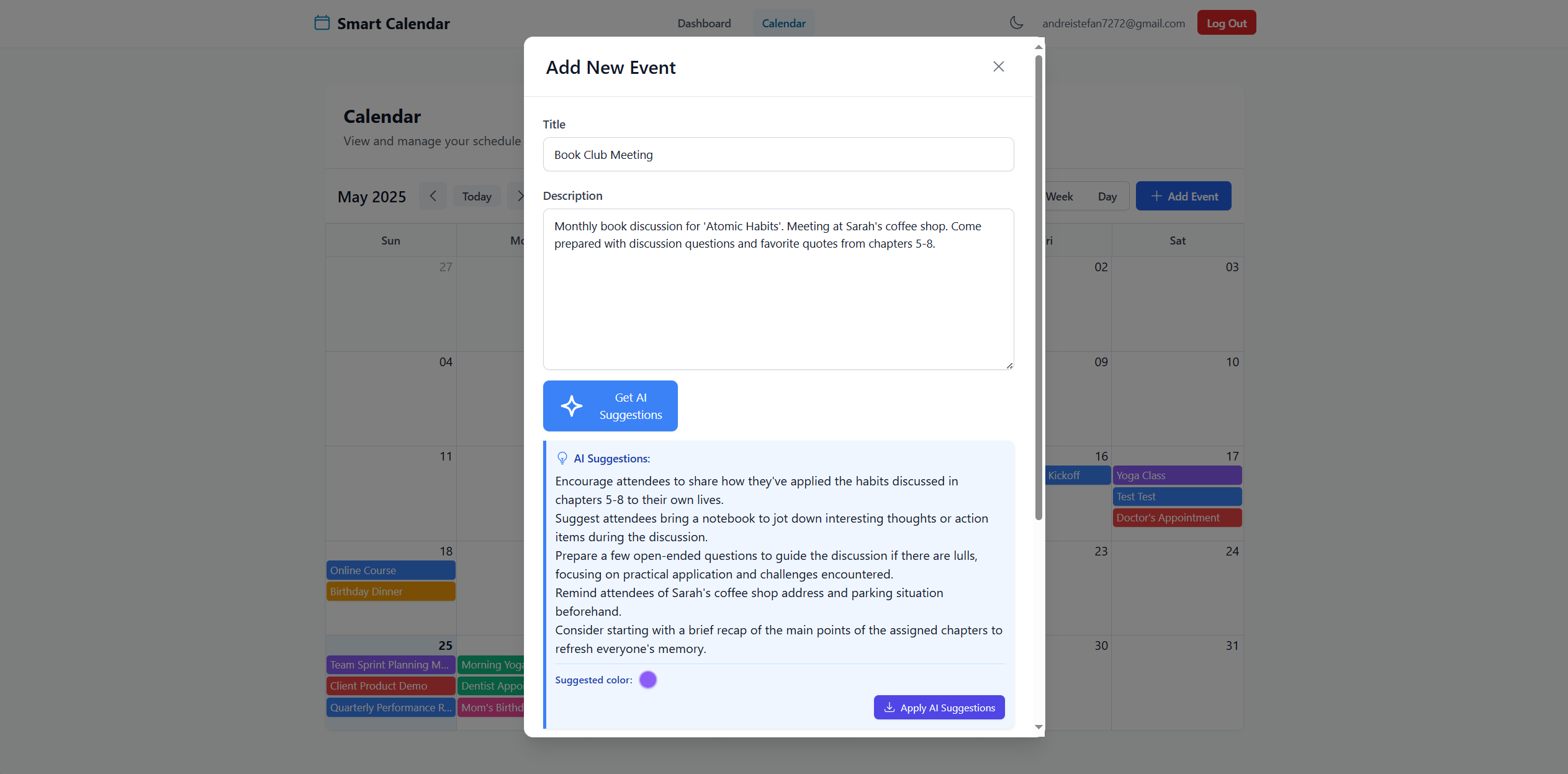
****

**5.3 Vizualizare Calendar**

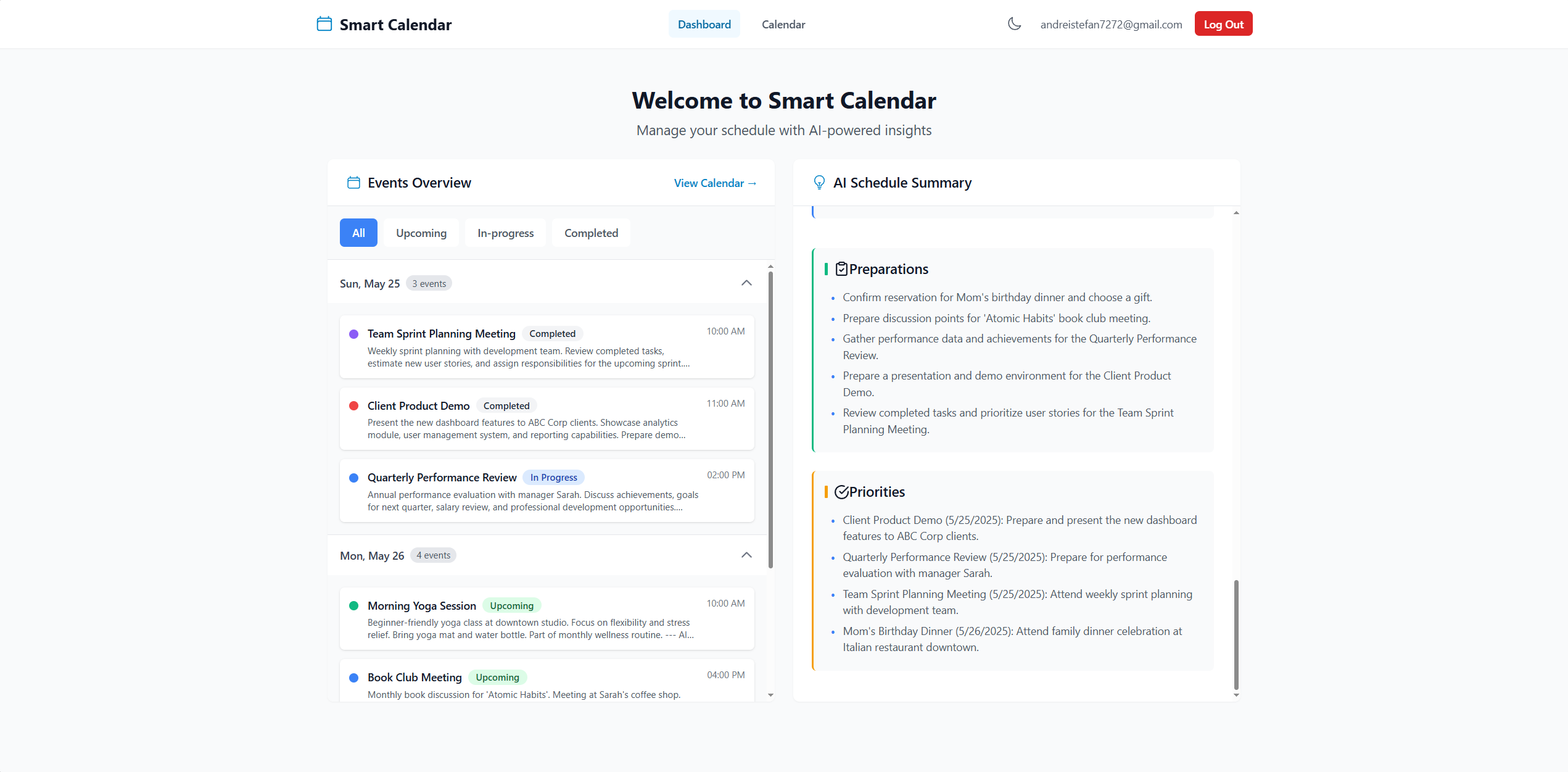
****

****

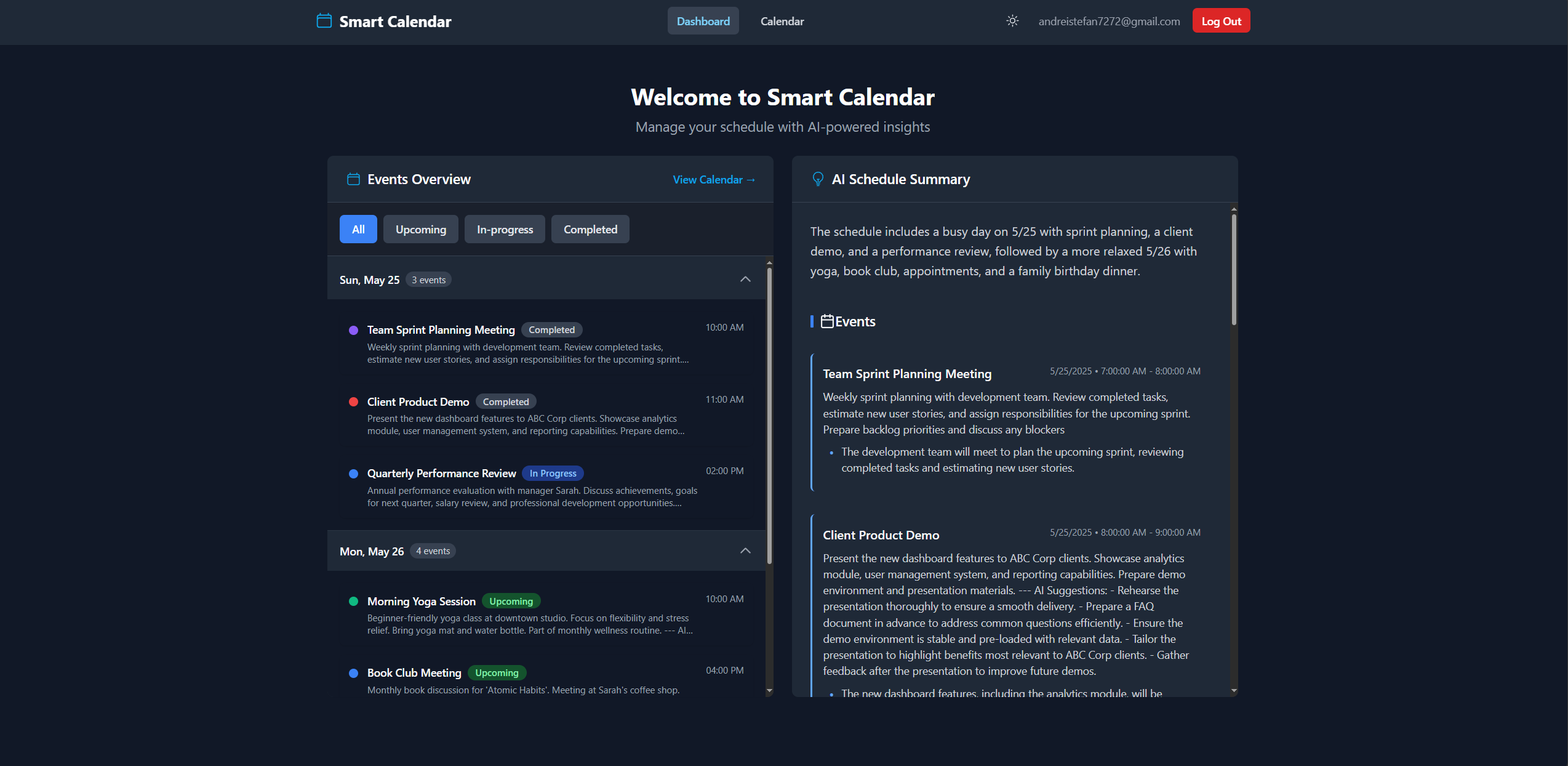
**5.4 Adăugare Eveniment cu AI**

****

**5.5 Rezumat Inteligent**

****

**5.6 Mod Întunecat**



**Caracteristici Tehnice**

**Frontend**: React 19, React Router 7, React Big Calendar, Tailwind CSS, Firebase Auth **Backend**: Express.js, Google Gemini AI (2.0), CORS **Cloud**: Firebase (Auth + Firestore), Google Generative AI **Styling**: Tailwind CSS cu suport dark mode **Build**: Vite pentru development și build optimizat

**Instalare și Configurare**

**Cerințe preliminare**

* Node.js 18+
* Cont Firebase
* Cheie API Google Gemini

**Pași instalare**

# Clonează repository-ul

git clone https://github.com/anro772/smart-calendar.git

cd smart-calendar

# Backend

cd server

npm install

# Creează .env cu PORT=5000 și GEMINI\_API\_KEY

# Frontend

cd ../smart-calendar-client

npm install

# Creează .env cu credențialele Firebase și VITE\_API\_URL=http://localhost:5000/api

# Pornire aplicație

npm run start # în server/

npm run start # în smart-calendar-client/

**Variabile de mediu necesare**

**Backend (.env)**:

PORT=5000

GEMINI\_API\_KEY=your\_gemini\_api\_key

**Frontend (.env)**:

VITE\_FIREBASE\_API\_KEY=your\_firebase\_api\_key

VITE\_FIREBASE\_AUTH\_DOMAIN=your\_project.firebaseapp.com

VITE\_FIREBASE\_PROJECT\_ID=your\_project\_id

VITE\_FIREBASE\_STORAGE\_BUCKET=your\_project.appspot.com

VITE\_FIREBASE\_MESSAGING\_SENDER\_ID=your\_sender\_id

VITE\_FIREBASE\_APP\_ID=your\_app\_id

VITE\_API\_URL=http://localhost:5000/api

**Structura Proiectului**

smart-calendar/

├── smart-calendar-client/ # Frontend React

│ ├── public/ # Fișiere publice

│ ├── src/

│ │ ├── components/ # Componente reutilizabile

│ │ ├── contexts/ # Context providers

│ │ ├── pages/ # Pagini aplicație

│ │ ├── services/ # Servicii API

│ │ │ └── aiService.js # Serviciu pentru AI

│ │ ├── styles/ # Stiluri CSS

│ │ ├── App.jsx # Componenta principală

│ │ ├── firebase.js # Configurare Firebase

│ │ └── main.jsx # Entry point

│ ├── .env # Variabile de mediu

│ └── package.json

├── server/ # Backend Express

│ ├── index.js # Server principal

│ ├── .env # Variabile de mediu server

│ └── package.json

└── README.md # Documentația

**Publicare și Deployment**

**Opțiuni recomandate**:

* **Frontend and Backend**:Render (cloud application platform)
* **Database**: Firebase Firestore (cloud-native)

**Dezvoltare Ulterioară**

* Integrare Google Calendar și Outlook
* Notificări push pentru evenimente
* Sharing și colaborare pe calendar
* Recurență avansată pentru evenimente
* Export/import în diverse formate

**6. Referințe**

1. **React Documentation** - <https://react.dev/>
2. **Express.js Guide** - <https://expressjs.com/>
3. **Google Generative AI** - <https://ai.google.dev/>
4. **Firebase Documentation** - <https://firebase.google.com/docs>
5. **Tailwind CSS** - <https://tailwindcss.com/>