CORS and API URL Configuration Fix Summary

🎯 Problem Identified

1. **Backend CORS Issue**: CORS only allowed `localhost:5173`, blocking requests from `https://workigom.vercel.app`
2. **Frontend API URL Issue**: API configuration wasn't flexible enough to handle different URL formats

✅ Fixes Applied

1. Backend CORS Configuration (`backend/src/app.ts`)

Before:

```
const corsOptions = {
  origin: process.env.CORS_ORIGIN || 'http://localhost:5173',
  credentials: true,
  optionsSuccessStatus: 200
};
```

After:

```
const allowedOrigins = [
  'http://localhost:5173',
  'https://workigom.vercel.app',
  process.env.CORS_ORIGIN
].filter(Boolean); // Remove undefined/null values

const corsOptions = {
  origin: (origin: string | undefined, callback: (err: Error | null, allow?: boolean)
=> void) => {
    // Allow requests with no origin (like mobile apps, curl, postman)
    if (!origin) return callback(null, true);

    if (allowedOrigins.indexOf(origin) !== -1) {
      callback(null, true);
    } else {
      console.warn(`CORS: Blocked origin ${origin}`);
      callback(new Error('Not allowed by CORS'));
    }
  },
  credentials: true,
  optionsSuccessStatus: 200
};
```

Benefits:
- ✅ Allows requests from both localhost and production frontend
- ✅ Supports additional origins via `CORS_ORIGIN` environment variable

- ✅ Provides clear logging when origins are blocked
- ✅ Allows requests with no origin (for testing tools)

---

2. Frontend API URL Configuration (`src-frontend/lib/api.ts`)

Before:

```
const API_URL = import.meta.env.VITE_BACKEND_URL || 'http://localhost:3001/api';
```

After:

```
// Construct API URL - handle both with and without /api suffix
const getApiUrl = () => {
  const backendUrl = import.meta.env.VITE_BACKEND_URL || 'http://localhost:3001';
  const baseUrl = backendUrl.endsWith('/api') ? backendUrl : `${backendUrl}/api`;

  // Debug logging in development
  if (import.meta.env.DEV) {
    console.log('🔧 API Configuration:', {
      VITE_BACKEND_URL: import.meta.env.VITE_BACKEND_URL,
      finalApiUrl: baseUrl,
      mode: import.meta.env.MODE
    });
  }

  return baseUrl;
};

const API_URL = getApiUrl();
```

Benefits:
- ✅ Automatically adds `/api` suffix if not present
- ✅ Works with both `https://workigom.onrender.com` and `https://workigom.onrender.com/api`
- ✅ Provides debug logging in development mode
- ✅ More flexible for different deployment scenarios

---

3. Axios Credentials Configuration

Added:

```
const api: AxiosInstance = axios.create({
  baseURL: API_URL,
  withCredentials: true, // Send cookies with requests
  headers: {
    'Content-Type': 'application/json',
  },
});
```

Benefits:
- ✅ Enables sending cookies with cross-origin requests
- ✅ Matches the backend CORS `credentials: true` setting

---

4. Environment Variable Documentation

Updated `src-frontend/.env.example`:

```
# Backend API URL
# For local development:
# VITE_BACKEND_URL=http://localhost:3001
#
# For production deployment (Vercel):
# Set this as an environment variable in Vercel dashboard
# VITE_BACKEND_URL=https://workigom.onrender.com
#
# Note: The /api suffix is automatically added, so don't include it
VITE_BACKEND_URL=http://localhost:3001
```

Updated `.gitignore`:

```
**/.env
**/.env.local
```

Benefits:
- ✅ Clear instructions for local and production setup
- ✅ Ensures `.env` files in subdirectories are ignored by git

🚀 Deployment Instructions for Vercel

Verify Environment Variable on Vercel:

1. Go to your Vercel project settings
2. Navigate to **Environment Variables**
3. Ensure you have:
   `VITE_BACKEND_URL=https://workigom.onrender.com`
   Note: Do NOT include `/api` - it will be added automatically
4. **Important:** Redeploy your frontend after these changes are pushed to trigger a new build

To Redeploy on Vercel:

Option 1: Automatic (Recommended)
- The changes are now pushed to your GitHub repository
- Vercel will automatically detect the push and trigger a new deployment
- Wait for the deployment to complete

Option 2: Manual
1. Go to your Vercel dashboard
2. Navigate to your `workigom` project
3. Go to the **Deployments** tab
4. Click on the three dots (…) next to the latest deployment
5. Click **Redeploy**
6. Select “Use existing Build Cache” → **No** (to ensure fresh build)

🔍 Testing After Deployment

1. Check Browser Console

Open https://workigom.vercel.app and check the browser console. You should see:

```
🔧 API Configuration: {
  VITE_BACKEND_URL: "https://workigom.onrender.com",
  finalApiUrl: "https://workigom.onrender.com/api",
  mode: "production"
}
```

2. Verify API Calls

- Open the **Network** tab in browser DevTools
- Try to login or make any API request
- Check that requests are going to `https://workigom.onrender.com/api/*`
- Verify there are no CORS errors in the console

3. Check Backend Logs

On Render.com, check your backend logs. You should NOT see any:
- `CORS: Blocked origin https://workigom.vercel.app` messages

📋 Files Modified

1. ✅ `backend/src/app.ts` - Updated CORS configuration
2. ✅ `src-frontend/lib/api.ts` - Improved API URL handling
3. ✅ `src-frontend/.env.example` - Updated documentation
4. ✅ `.gitignore` - Added subdirectory .env files
5. ✅ `src-frontend/.env` - Updated local development config

🎉 Expected Results

After deployment:
- ✅ Frontend on Vercel can successfully communicate with backend on Render
- ✅ No CORS errors in browser console
- ✅ All API requests use production backend URL
- ✅ Authentication and data fetching work correctly

🐛 Troubleshooting

If frontend still uses localhost:

1. **Clear Vercel build cache:**
   - Go to Vercel dashboard → Settings → General
   - Scroll to "Clear Build Cache & Redeploy"

2. **Verify environment variable format:**
   - Should be: `VITE_BACKEND_URL=https://workigom.onrender.com`
   - NOT: `VITE_BACKEND_URL=https://workigom.onrender.com/api`

3. **Check Vercel build logs:**
   - Look for "Environment Variables" section
   - Ensure `VITE_BACKEND_URL` is listed

If CORS errors persist:

1. **Check backend deployment:**
   - Ensure the latest changes are deployed on Render
   - Check backend logs for CORS-related messages

2. **Verify the origin:**
   - Make sure you're accessing via `https://workigom.vercel.app`
   - Not via a preview URL like `workigom-*.vercel.app`
   - If using preview URLs, add them to the CORS config

📝 Commit Information

Commit Hash: `9a2479e`
Commit Message: "fix: Update CORS config and frontend API URL handling"
Pushed to: `master` branch on GitHub

✨ Next Steps

1. ✅ Changes are pushed to GitHub
2. 🔄 Wait for Vercel to auto-deploy (or trigger manual deployment)
3. 🧪 Test the application at https://workigom.vercel.app
4. 📊 Monitor browser console and network requests
5. 🎯 Verify all features work correctly

Generated: October 24, 2025