

Projektplan: Offene Punkte - SENECDesign Website

Projekt: leipzig-photovoltaik.de (SENEC-Design)

Status: Hauptfunktionen abgeschlossen (Mobile PageSpeed ⁹⁷/₁₀₀)





Erstellt: 30. Januar 2026

Version: 1.0

Übersicht

Dieser Plan listet alle **optionalen/späteren** Aufgaben mit Prioritäten, Zeitschätzungen und Abhängigkeiten auf.

Legende:

-  **Kritisch** - Blockiert andere Aufgaben oder essentiell für Launch
 -  **Hoch** - Wichtig für SEO/Conversion, sollte bald erledigt werden
 -  **Mittel** - Nice-to-have, verbessert User Experience
 -  **Niedrig** - Optional, kann später gemacht werden
-



Prioritäten-Matrix

Kategorie	Kritisch	Hoch	Mittel	Niedrig	Gesamt
Deployment	1	1	-	-	2
SEO	-	3	1	-	4
Marketing	-	1	1	-	2
Content	-	-	1	-	1
Features	-	-	2	2	4
Monitoring	-	-	1	2	3
Gesamt	1	5	6	4	16

● KRITISCH (Priorität 1)

1. Code zu GitHub pushen

Kategorie: Deployment

Priorität: ● Kritisch

Dauer: 10 Minuten

Abhängigkeiten: Keine

Beschreibung: Code ist lokal committed (Commit 909379d6), muss zu GitHub gepusht werden für Vercel-Deployment.

Schritte:

1. GitHub-Repository erstellen (falls noch nicht vorhanden)
2. Remote hinzufügen: `git remote add origin https://github.com/USERNAME/senec-design.git`
3. Push durchführen: `git push -u origin main`
4. Vercel mit GitHub-Repo verbinden

Blocker:

- ❌ Vercel-Deployment nicht möglich ohne GitHub-Repo

Erfolgskriterien:

- ☒ Code auf GitHub sichtbar
- ☒ Vercel kann Repo lesen
- ☒ Automatisches Deployment bei Push funktioniert

Ressourcen:

- Anleitung: `/home/ubuntu/senec-design/GITHUB-PUSH-ANLEITUNG.md`
-

● HOCH (Priorität 2)

2. URL-Struktur an leipzig-photovoltaik.de anpassen

Kategorie: SEO

Priorität: ● Hoch

Dauer: 2-3 Stunden

Abhängigkeiten: Keine

Beschreibung: Aktuelle URL-Struktur weicht von alter Website ab. Für SEO-Kontinuität sollten URLs identisch sein.

Analyse:

Alte Website (leipzig-photovoltaik.de):

- /photovoltaik-leipzig
- /photovoltaik-dresden
- /photovoltaik-halle
- /blog/artikel-slug

Neue Website (aktuell):

- /standorte/leipzig
- /standorte/dresden
- /standorte/halle
- /blog/artikel-slug (☒ identisch)

Schritte:

1. Redirects in `vercel.json` oder `.htaccess` einrichten
2. Alte URLs → Neue URLs (301 Permanent Redirect)

3. Oder: Seiten umbenennen und interne Links anpassen
4. Google Search Console über Änderungen informieren

Empfehlung:

- **Option A:** Redirects einrichten (schneller, SEO-sicher)
- **Option B:** Seiten umbenennen (sauberer, mehr Aufwand)

Blocker:

- ⚠️ Potentieller SEO-Ranking-Verlust wenn nicht gemacht

Erfolgskriterien:

- ✅ Alle alten URLs funktionieren (mit 301 Redirect)
 - ✅ Google Search Console zeigt keine 404-Fehler
 - ✅ Rankings bleiben stabil
-

3. Google Place ID für Reviews Widget

Kategorie: Marketing

Priorität: 🟡 Hoch

Dauer: 30 Minuten

Abhängigkeiten: Google My Business Account

Beschreibung: Reviews Widget ist implementiert, aber zeigt nur Fallback-Bewertungen. Echte Google-Bewertungen benötigen Place ID.

Schritte:

1. Google My Business Account öffnen
2. Place ID finden:
 - Option A:
<https://developers.google.com/maps/documentation/javascript/examples/places-placeid-finder>
 - Option B: Google Maps → Geschäft suchen → URL kopieren → Place ID extrahieren
3. Place ID in `GoogleReviews.astro` eintragen
4. Build testen
5. Live-Website verifizieren

Place ID Format:

ChIJ... (ca. 27 Zeichen)

Blocker:

- ⚠ Ohne Place ID keine echten Bewertungen sichtbar

Erfolgskriterien:

- ☒ Echte Google-Bewertungen auf Homepage sichtbar
 - ☒ Sternebewertung korrekt angezeigt
 - ☒ Bewertungstexte laden
-

4. Google Rich Results Test

Kategorie: SEO

Priorität: ● Hoch

Dauer: 1-2 Stunden

Abhängigkeiten: Website live

Beschreibung: Prüfen ob strukturierte Daten (Schema.org) korrekt implementiert sind für Rich Snippets in Google.

Zu testen:

- LocalBusiness Schema
- FAQ Schema
- Article Schema (Blog-Artikel)
- BreadcrumbList Schema
- Review Schema (wenn Reviews aktiv)

Schritte:

1. Website zu Google Rich Results Test hinzufügen
2. Alle Seiten einzeln testen
3. Fehler beheben (falls vorhanden)
4. Google Search Console validieren

Tools:

- <https://search.google.com/test/rich-results>
- <https://validator.schema.org>

Blocker:

- ⚠ Fehlende Rich Snippets = niedrigere Click-Through-Rate

Erfolgskriterien:

- ☒ Alle Seiten ohne Schema-Fehler
 - ☒ Rich Snippets in Google-Vorschau sichtbar
 - ☒ FAQ-Snippets in Suchergebnissen
-

5. Sitemap bei Google Search Console einreichen

Kategorie: SEO

Priorität:  Hoch

Dauer: 15 Minuten

Abhängigkeiten: Website live, Google Search Console Account

Beschreibung: Sitemap existiert (`/sitemap-index.xml`), muss bei Google eingereicht werden für schnellere Indexierung.

Schritte:

1. Google Search Console öffnen
2. Property hinzufügen (falls noch nicht vorhanden)
3. Sitemap einreichen: `https://senecdesign-frkgze1v.manus.space/sitemap-index.xml`
4. Indexierungsstatus überwachen

Blocker:

- ⚠ Langsamere Indexierung ohne Sitemap-Einreichung

Erfolgskriterien:

- ☒ Sitemap erfolgreich eingereicht
- ☒ Keine Fehler in Search Console
- ☒ Seiten werden indexiert

6. robots.txt optimieren

Kategorie: SEO

Priorität: 🟡 Hoch

Dauer: 15 Minuten

Abhängigkeiten: Keine

Beschreibung: robots.txt prüfen und optimieren für Crawler-Zugriff.

Zu prüfen:

- Sitemap-Verweis vorhanden?
- Admin-Bereiche blockiert?
- Wichtige Seiten nicht versehentlich blockiert?

Schritte:

1. /public/robots.txt prüfen
2. Falls nicht vorhanden: erstellen
3. Sitemap-URL hinzufügen
4. Build testen

Beispiel:

```
User-agent: *  
Allow: /  
  
Sitemap: https://senecdesign-frkgzlv.manus.space/sitemap-index.xml
```

Erfolgskriterien:

- ✅ robots.txt erreichbar unter /robots.txt
 - ✅ Sitemap-Verweis vorhanden
 - ✅ Keine wichtigen Seiten blockiert
-

MITTEL (Priorität 3)

7. Referenzen-Seite ausbauen

Kategorie: Content

Priorität:  Mittel

Dauer: 3-4 Stunden (abhängig von User-Input)

Abhängigkeiten: User stellt Referenz-Daten bereit

Beschreibung: Referenzen-Seite existiert, aber ohne echte Kundenprojekte. Wartet auf User-Input.

Benötigte Informationen pro Referenz:

- Kundenname (oder anonymisiert: "Einfamilienhaus Leipzig")
- Standort
- Anlagengröße (kWp)
- Installationsdatum
- Vorher/Nachher-Bilder (optional)
- Einsparungen (€/Jahr, CO₂)
- Kundenzitat (optional)




Schritte:

1. User stellt Referenz-Daten bereit
2. Referenz-Komponente erstellen
3. Bilder optimieren (WebP)
4. SEO-Texte schreiben
5. Build testen

Blocker:

-  Wartet auf User-Input

Erfolgskriterien:

-  Mindestens 3-5 Referenzen sichtbar
 -  Bilder in WebP-Format
 -  SEO-optimierte Beschreibungen
-

8. Lead-Tracking-System implementieren

Kategorie: Features

Priorität: 🟡 Mittel

Dauer: 4-6 Stunden

Abhängigkeiten: Datenbank-Setup

Beschreibung: Lead-Zähler mit Tracking von:

- Datum der Anfrage
- Name des Anfragenden
- Status: Danke-E-Mail versendet?
- Status: Double-Opt-In bestätigt?

Technische Umsetzung:

- Option A: Google Sheets API (einfach, keine DB nötig)
- Option B: Airtable API (strukturierter)
- Option C: Eigene Datenbank (komplexer, mehr Kontrolle)

Schritte:

1. Technologie wählen
2. API-Integration implementieren
3. Formular-Handler erweitern
4. Dashboard erstellen (optional)
5. DSGVO-Konformität prüfen

Blocker:

- ⚠️ Benötigt Entscheidung: Welche Technologie?

Erfolgskriterien:

- ✅ Leads werden gespeichert
 - ✅ Status-Tracking funktioniert
 - ✅ DSGVO-konform
-

9. Performance-Monitoring einrichten

Kategorie: Monitoring

Priorität: 🟡 Mittel

Dauer: 2-3 Stunden

Abhängigkeiten: Website live

Beschreibung: Automatisches Monitoring von PageSpeed, Uptime und Fehlern.

Tools (kostenlos):

- **Uptime:** UptimeRobot, Pingdom
- **PageSpeed:** Google PageSpeed Insights API
- **Fehler:** Sentry (Free Tier)

Schritte:

1. Tools auswählen
2. Accounts erstellen
3. Monitoring einrichten
4. Alerts konfigurieren (E-Mail bei Ausfall)
5. Wöchentliche Reports aktivieren

Erfolgskriterien:

- ☒ Uptime-Monitoring aktiv
 - ☒ PageSpeed-Tracking läuft
 - ☒ Fehler-Alerts funktionieren
-

10. Conversion-Tracking (Google Analytics Events)

Kategorie: Marketing

Priorität: 🟡 Mittel

Dauer: 1-2 Stunden

Abhängigkeiten: Google Analytics aktiv

Beschreibung: Event-Tracking für wichtige User-Aktionen:

- “Angebot anfordern” Button-Klick
- Formular-Absendung

- Telefon-Klick
- E-Mail-Klick
- Solarrechner-Nutzung




Schritte:

1. Google Analytics Events definieren
2. Event-Tracking-Code hinzufügen
3. Google Tag Manager einrichten (optional)
4. Events in Google Analytics testen
5. Conversion-Ziele erstellen

Code-Beispiel:

```
// Button-Klick tracken
gtag('event', 'button_click', {
  'event_category': 'engagement',
  'event_label': 'angebot_anfordern'
});
```

Erfolgskriterien:

-  Events in Google Analytics sichtbar
-  Conversion-Rate messbar
-  Funnel-Analyse möglich

11. Broken-Link-Checker einrichten

Kategorie: Monitoring

Priorität:  Mittel

Dauer: 1 Stunde

Abhängigkeiten: Website live

Beschreibung: Automatische Prüfung auf defekte Links (intern und extern).

Tools:

- **Online:** Dead Link Checker, Broken Link Check
- **CLI:** `linkchecker` (Python)

- **CI/CD:** GitHub Actions mit linkchecker

Schritte:

1. Tool auswählen
2. Initiale Prüfung durchführen
3. Gefundene Fehler beheben
4. Automatische wöchentliche Prüfung einrichten

Erfolgskriterien:

- ☒ Keine defekten internen Links
 - ☒ Externe Links funktionieren
 - ☒ Automatische Prüfung läuft
-

NIEDRIG (Priorität 4)

12. Admin-Dashboard erstellen

Kategorie: Features

Priorität:  Niedrig

Dauer: 8-12 Stunden

Abhängigkeiten: Authentifizierung

Beschreibung: Internes Dashboard für:

- Lead-Übersicht
- Performance-Metriken
- Content-Management
- SEO-Reports

Technische Umsetzung:

- Basic Auth oder Passwort-Sperre
- Dashboard-Layout mit Navigation
- Integration mit Lead-Tracking
- Performance-Daten von Google Analytics API

Schritte:

1. Authentifizierung implementieren
2. Dashboard-Layout erstellen
3. Datenquellen anbinden
4. Reports generieren
5. Sicherheit prüfen

Blocker:

- ⚠ Benötigt Lead-Tracking-System (Task #8)

Erfolgskriterien:

- ✓ Dashboard erreichbar unter `/admin`
 - ✓ Passwort-Schutz funktioniert
 - ✓ Daten werden korrekt angezeigt
-

13. SEO-Analyse-Tool integrieren

Kategorie: Monitoring

Priorität: ● Niedrig

Dauer: 3-4 Stunden

Abhängigkeiten: Keine

Beschreibung: Tool zur automatischen SEO-Analyse:

- Meta-Tags prüfen
- Heading-Struktur validieren
- Bild-Alt-Texte checken
- Sitemap-Status

Tools:

- Lighthouse CI
- SEO-Checker-APIs
- Custom Script

Schritte:

1. Tool auswählen
2. Integration implementieren

3. Automatische Prüfung bei Deployment

4. Reports generieren

Erfolgskriterien:

- ☒ SEO-Score wird automatisch berechnet
 - ☒ Fehler werden gemeldet
 - ☒ Verbesserungsvorschläge verfügbar
-

14. Pre-Deployment-Checks automatisieren

Kategorie: Features

Priorität: ● Niedrig

Dauer: 2-3 Stunden

Abhängigkeiten: CI/CD-Pipeline

Beschreibung: Automatische Prüfungen vor jedem Deployment:

- Build erfolgreich?
- Keine Broken Links?
- PageSpeed > 85?
- SEO-Score > 90?

Technische Umsetzung:


- GitHub Actions Workflow
- Pre-Deployment Hooks
- Vercel Build Hooks

Schritte:

1. GitHub Actions Workflow erstellen
2. Checks definieren
3. Deployment blockieren bei Fehlern
4. Notifications einrichten

Erfolgskriterien:

- ☒ Checks laufen automatisch
- ☒ Deployment wird bei Fehlern blockiert

-  Team wird benachrichtigt
-

15. fontkit windows-1252 encoding Fehler beheben

Kategorie: Bugfix

Priorität:  Niedrig

Dauer: 2-4 Stunden (Recherche + Fix)

Abhängigkeiten: Keine

Beschreibung: Bekannter Fehler in Astro 5.15.9 (auch nach Downgrade vorhanden). Aktuell kein Blocker, aber sollte langfristig behoben werden.

Mögliche Lösungen:

1. Warten auf Astro-Update
2. Alternative PDF-Library verwenden
3. Workaround implementieren



Schritte:

1. GitHub Issues prüfen
2. Alternative Libraries testen
3. Fix implementieren
4. Build verifizieren

Blocker:

-  Kein akuter Blocker, Website funktioniert

Erfolgskriterien:

-  Build ohne Encoding-Warnung
 -  Alle Funktionen intakt
-

16. Wöchentlicher Automatisierungs-Prompt

Kategorie: Automation

Priorität:  Niedrig

Dauer: 1-2 Stunden

Abhängigkeiten: Keine

Beschreibung: Automatisierter wöchentlicher Prompt für:

- Projekt-Status-Update
- Offene Tasks priorisieren
- Neue Optimierungen vorschlagen
- Performance-Report

Technische Umsetzung:

- Scheduled Task in Manus
- Wöchentliche Erinnerung
- Template-Prompt

Schritte:

1. Prompt-Template erstellen
2. Schedule einrichten (jeden Montag 9:00 Uhr)
3. Automatische Ausführung testen

Erfolgskriterien:

- ☒ Wöchentlicher Prompt wird automatisch gestartet
- ☒ Projekt-Status wird aktualisiert
- ☒ Neue Tasks werden identifiziert



Zeitplan-Vorschlag

Phase 1: Kritisch & Deployment (Woche 1)

Dauer: 1-2 Tage

Aufwand: ~4 Stunden

1. ☒ Code zu GitHub pushen (10 Min)
2. ☒ Vercel-Deployment einrichten (30 Min)
3. ☒ Google Place ID hinzufügen (30 Min)
4. ☒ robots.txt optimieren (15 Min)
5. ☒ Sitemap einreichen (15 Min)

6. ☒ URL-Struktur anpassen (2-3 Std)

Ergebnis: Website ist live und SEO-optimiert

Phase 2: SEO & Marketing (Woche 2)

Dauer: 2-3 Tage

Aufwand: ~6 Stunden

1. ☒ Google Rich Results Test (1-2 Std)
2. ☒ Conversion-Tracking einrichten (1-2 Std)
3. ☒ Performance-Monitoring (2-3 Std)

Ergebnis: SEO vollständig optimiert, Tracking aktiv

Phase 3: Features & Content (Woche 3-4)

Dauer: 1-2 Wochen

Aufwand: ~12 Stunden

1. ☒ Lead-Tracking-System (4-6 Std)
2. ☒ Referenzen-Seite ausbauen (3-4 Std, wartet auf User)
3. ☒ Broken-Link-Checker (1 Std)

Ergebnis: Vollständige Marketing-Infrastruktur

Phase 4: Automation & Monitoring (Woche 5+)

















Dauer: Nach Bedarf

Aufwand: ~15 Stunden

1. ☒ Admin-Dashboard (8-12 Std)
2. ☒ SEO-Analyse-Tool (3-4 Std)
3. ☒ Pre-Deployment-Checks (2-3 Std)
4. ☒ Wöchentlicher Prompt (1-2 Std)
5. ☒ fontkit-Fehler beheben (2-4 Std)

Ergebnis: Vollautomatisierte Infrastruktur

Kosten-Nutzen-Analyse






Aufgabe	Aufwand	Impact	ROI	Empfehlung
GitHub Push	10 Min	 Kritisch	★★★★★★	Sofort
URL-Struktur	2-3 Std	 Hoch	★★★★★	Diese Woche
Google Place ID	30 Min	 Hoch	★★★★★★	Diese Woche
Rich Results	1-2 Std	 Hoch	★★★★★	Diese Woche
Sitemap	15 Min	 Hoch	★★★★★★	Diese Woche
robots.txt	15 Min	 Hoch	★★★★★★	Diese Woche
Lead-Tracking	4-6 Std	 Mittel	★★★★	Nächste Woche
Conversion-Tracking	1-2 Std	 Mittel	★★★★★	Nächste Woche
Performance-Monitoring	2-3 Std	 Mittel	★★★★	Nächste Woche
Referenzen	3-4 Std	 Mittel	★★★★	Bei Bedarf
Broken-Link-Checker	1 Std	 Mittel	★★★	Bei Bedarf
Admin-Dashboard	8-12 Std	 Niedrig	★★★	Später
SEO-Analyse-Tool	3-4 Std	 Niedrig	★★★	Später
Pre-Deployment-Checks	2-3 Std	 Niedrig	★★★	Später
fontkit-Fehler	2-4 Std	 Niedrig	★	Optional
Wöchentlicher Prompt	1-2 Std	 Niedrig	★★	Optional

ROI-Legende:

- ★★★★★★ Sehr hoch (wenig Aufwand, großer Nutzen)
- ★★★★★ Hoch
- ★★★★ Mittel
- ★★★ Niedrig
- ★ Sehr niedrig




Empfohlene Reihenfolge (Quick Wins zuerst)

Woche 1: Deployment & Quick Wins (4 Stunden)

1.  Code zu GitHub pushen (10 Min) → **SOFORT**
2.  robots.txt optimieren (15 Min)
3.  Sitemap einreichen (15 Min)
4.  Google Place ID (30 Min)
5.  URL-Struktur (2-3 Std)




Ergebnis: Website live, SEO-Basics erledigt, Reviews aktiv

Woche 2: SEO-Optimierung (6 Stunden)

1.  Google Rich Results Test (1-2 Std)
2.  Conversion-Tracking (1-2 Std)
3.  Performance-Monitoring (2-3 Std)





Ergebnis: Vollständige SEO-Optimierung, Tracking aktiv

Woche 3-4: Features (12 Stunden)

1.  Lead-Tracking-System (4-6 Std)
2.  Referenzen-Seite (3-4 Std)
3.  Broken-Link-Checker (1 Std)

Ergebnis: Marketing-Infrastruktur komplett

Später: Automation (15+ Stunden)

1.  Admin-Dashboard (8-12 Std)
2.  SEO-Analyse-Tool (3-4 Std)
3.  Pre-Deployment-Checks (2-3 Std)
4.  Weitere Optimierungen

Ergebnis: Vollautomatisierte Infrastruktur



Checkliste für Projektabschluss

Deployment

- ☐ Code auf GitHub
- ☐ Vercel-Deployment aktiv
- ☐ Custom Domain verbunden
- ☐ SSL-Zertifikat aktiv

SEO

- ☐ URL-Struktur optimiert
- ☐ robots.txt vorhanden
- ☐ Sitemap eingereicht
- ☐ Rich Results validiert
- ☐ Google Search Console eingerichtet
- ☐ Google Analytics Events tracken

Marketing

- ☐ Google Place ID hinterlegt
- ☐ Reviews Widget aktiv
- ☐ Conversion-Tracking läuft
- ☐ Lead-Tracking funktioniert

Content

- ☐ Referenzen-Seite ausgefüllt
- ☐ Alle Bilder optimiert (WebP)
- ☐ Blog-Artikel vollständig

Monitoring

- ☐ Performance-Monitoring aktiv

- ☐ Uptime-Monitoring läuft
- ☐ Broken-Link-Checker eingerichtet
- ☐ Fehler-Alerts konfiguriert

Optional

- ☐ Admin-Dashboard erstellt
- ☐ SEO-Analyse-Tool integriert
- ☐ Pre-Deployment-Checks automatisiert
- ☐ Wöchentlicher Prompt eingerichtet



Nächste Schritte

Empfehlung für diese Woche:

1. **SOFORT:** Code zu GitHub pushen (10 Min)
2. **Heute:** robots.txt + Sitemap + Google Place ID (1 Std)
3. **Diese Woche:** URL-Struktur anpassen (2-3 Std)
4. **Diese Woche:** Google Rich Results Test (1-2 Std)

Gesamtaufwand Woche 1: ~4-6 Stunden

Impact: Website live, SEO-optimiert, Reviews aktiv



Fragen?

Bei Unklarheiten oder Änderungswünschen:

- Projektplan kann jederzeit angepasst werden
- Prioritäten können verschoben werden
- Zeitschätzungen sind Richtwerte

Nächster Review: Nach Abschluss Phase 1 (Deployment)

Viel Erfolg! 