Technical Specification

1. Overview

ImpactCanvas is a user-friendly web application designed to streamline the generation of social impact project documentation. By capturing a simplified Social Canvas through a digital form, the application leverages AI to automatically produce:

- **Theory of Change** statements
- Recommended **Impact Activities**
- **Monitoring and Evaluation** (M&E) plans

---

2. System Architecture

ImpactCanvas follows a simple, modern, client-server architecture.

Frontend

- **Framework:** Next.js
- **Form Component:** SurveyJS (survey-react)
- **HTTP Client:** Axios for API calls
- **Output Display:** Editable text areas and structured tables for user refinement
- **Deployment:** Vercel, Netlify, or any similar static host provider

Backend

- **Framework:** Python (FastAPI recommended)
- **Language Model Integration:** OpenAI GPT-4 API
- **Data Format:** JSON (structured based on SurveyJS schema)
- **Error Handling:** HTTP error codes and descriptive error responses
- **Deployment:** Docker containers, cloud serverless (e.g., AWS Lambda), or Supabase Edge Functions

Database and Storage

- **Recommended:** PostgreSQL via Supabase (fast integration)
- **Alternatives:** Firebase, MongoDB Atlas, JSON file storage (for prototyping)

---

3. Workflow (Customer’s Perspective)

```
---
config:
  theme: neo
---
flowchart LR
  A([Start]) --> B[Opens Web App]
  B --> C[Fills out SurveyJS Form<br/> Simplified Theory of Change]
  C --> D[Reviews & Submits Form]
  D -->|JSON Data| E[Backend receives submission]
  E --> F[AI generates outputs:<br/>- Theory of Change<br/>- Impact Activitie
s<br/>- Monitoring Plan]
  F --> G[Outputs returned to customer]
  G --> H[Customer reviews generated outputs]
  H --> I{Satisfied?}
  I -->|Yes| J[Exports & Downloads Report]
  I -->|No| K[Edits outputs directly in Web App]
  K --> J
  J --> L[Process Complete]
```

---

4. Frontend Implementation

SurveyJS Form

- Captures structured project information:
- Purpose, Vision, Mission, Approach
- Key Activities (up to 3)
- Resources (human, physical, financial, partnerships)

- Audience (beneficiaries, customers)
- Indicators for Monitoring

Frontend Responsibilities

- Displaying SurveyJS form clearly and responsively.
- Validating inputs (required fields).
- Submitting JSON data via Axios to the backend API.
- Displaying and enabling edits on the AI-generated outputs.

---

5. Backend Implementation

FastAPI Endpoint Specification

- **URL:** /generate-theory-of-change/
- **Method:** POST
- **Request Content-Type:** application/json

Request Body Example (from SurveyJS):

```
{
  "purpose": "To address food insecurity",
  "vision": "A resilient community garden",
  "mission": "...",
  "approach": "...",
  "activity1": "...",
  "activity2": "...",
  "activity3": "...",
  "humanResources": "...",
  "physicalResources": "...",
  "financialResources": "...",
  "partnerships": "...",
  "beneficiaries": "...",
  "customers": "...",
  "indicator1": "...",
```

```
  "indicator2": "...",
  "indicator3": "..."
}
```

Response:

```
{
  "generated_output": "Theory of Change:...\n\nImpact Activities:\n-...\n-...\n\n
Monitoring Plan:\n| Indicator | Method | Frequency |\n|-----------|--------|----
-------|\n| ...       | ...    | ...       |"
}
```

AI Integration

- **Prompt:** Carefully crafted prompts to GPT-4 using the structured JSON from SurveyJS.
- **AI Model:** GPT-4 (gpt-4-turbo recommended)
- **API Response Handling:** Includes basic error management, retries, and timeout logic.

---

6. Security and Privacy

- **Authentication/Authorization:** (Optional MVP) Consider integrating simple OAuth (Google/GitHub) if multi-user access needed.
- **Data Encryption:** HTTPS enforced for all data transfers.
- **API Keys and Sensitive Data:** Environment variables (.env), not hardcoded.

---

7. Deployment and Infrastructure

Recommended Deployment Stack (simplified):

- **Frontend:** Netlify, Vercel, GitHub Pages
- **Backend/API:** Docker container or serverless functions (AWS Lambda, Vercel Serverless, Supabase Edge Functions)
- **Database:** Supabase (PostgreSQL), Firebase, or MongoDB Atlas

- **Monitoring/Observability:** Sentry, Grafana, Prometheus (for production readiness)

8. Extensibility and Future Features

Potential future enhancements include:

- Integration with existing CRM or M&E systems (e.g., Salesforce Nonprofit Cloud, Airtable).
- User accounts and project history storage.
- Detailed reporting exports (PDF, DOCX, Markdown).
- Community feedback and learning system for continuous AI improvement.

9. Technology Stack Summary

Component	Technology/Framework
Frontend	Next.js, SurveyJS, Axios
Backend	Python (FastAPI)
AI Generation	GPT-4 via OpenAI API
Database	Supabase (PostgreSQL)
Storage	Cloud Storage or Supabase
Deployment	Vercel, Netlify, AWS Lambda
Security	HTTPS, OAuth (Optional)
Observability & Monitoring	Sentry, Grafana, Prometheus

10. Immediate Next Steps

- Implement minimal Next.js app with SurveyJS form and Axios.
- Set up and run backend FastAPI service.
- Conduct integration testing and validate outputs.
- Deploy minimal viable solution to a cloud host for user testing.