Product Requirements and Specification Document

Project Name

TravelSafe - AI Trip Risk Checker

Description

TravelSafe is a web application enabling users to input travel destinations and receive AI-driven risk assessments based on recent data. The app demonstrates data input, risk prediction, and practical travel safety applications.

1. Goals & Objectives

Goal	Description
User Safety Awareness	Inform users of potential travel risks for selected destinations
Practical ML Application	Demonstrate real-world use of machine learning in travel safety
Simple, Intuitive Experience	Ensure ease of use for all users

2. Core Features

Feature	Description
Destination Input	Users enter a city/country as their travel destination
Risk Prediction	ML model predicts risk level (Low/Medium/High) based on recent data
Risk Explanation	Brief summary of key risk factors (e.g., crime, health, weather)
Recent Data Integration	Utilizes up-to-date data sources for risk assessment
Responsive UI	Clean, modern, and mobile-friendly interface

3. User Stories

As a…	I want to…	So that…
Traveler	Enter a destination	I can check its current travel risks
Traveler	See a clear risk level and explanation	I understand why a destination is risky
Educator	Demonstrate ML in action	I can teach practical AI applications

4. Functional Requirements

ID	Requirement
FR1	Users can input a travel destination (city/country)
FR2	System fetches recent data for the destination
FR3	ML model predicts and returns risk level
FR4	System displays risk level and brief explanation
FR5	UI is responsive and accessible

---

5. Non-Functional Requirements

ID	Requirement
NFR1	Response time < 2 seconds per request
NFR2	System uptime ≥ 99%
NFR3	Data privacy: No user data stored
NFR4	Codebase is well-documented and modular

---

6. Technical Specifications

6.1 Architecture

- **Frontend:** HTML/CSS/JS (Flask templates)
- **Backend:** Python (Flask)
- **ML Model:** Pre-trained or simple classifier (e.g., scikit-learn)
- **Data Sources:** Public APIs or static datasets (e.g., crime, health, weather)
- **Deployment:** Local or cloud (Heroku, Render, etc.)

6.2 Data Flow

```
graph TD
A[User Input] --> B[Flask Backend]
B --> C[Fetch Recent Data]
C --> D[ML Risk Prediction]
D --> E[Return Risk Level & Explanation]
E --> F[Display to User]
```

6.3 API Endpoints

Endpoint	Method	Description	Request Params	Response
`/`	GET	Home page	-	HTML
`/check-risk`	POST	Check risk for destination	`destination` (str)	`{risk_level, explanation}`

---

7. UI/UX Requirements

Requirement	Description
Modern, futuristic theme	Clean, minimal, visually appealing
Simple input form	Single field for destination
Clear risk display	Prominent risk level, concise explanation
Mobile-friendly	Responsive layout for all devices

---

8. Acceptance Criteria

ID	Criteria
AC1	User can input a destination and receive a risk assessment
AC2	Risk level and explanation are accurate and understandable
AC3	System responds within 2 seconds
AC4	UI is accessible and works on mobile devices

---

9. Out of Scope

- User authentication or profiles
- Real-time data scraping
- Multi-language support
- Advanced ML model tuning

---

10. Milestones & Timeline

Milestone	Description	Target Date
Requirements Finalized	PRD approved	Day 1
ML Model Ready	Data collection & model training	Day 3
Backend/API Complete	Flask endpoints implemented	Day 5
Frontend Complete	UI/UX implemented	Day 7
Testing & Deployment	QA and go-live	Day 8

---

11. Risks & Mitigations

Risk	Mitigation
Data source unavailability	Use static fallback datasets
ML model inaccuracy	Use simple, interpretable models

UI/UX complexity	Keep design minimal and focused

---

12. Appendix

Example Risk Prediction Output

```
{
  "destination": "Paris, France",
  "risk_level": "Medium",
  "explanation": "Recent increase in petty crime and moderate COVID-19 cases."
}
```

End of Document