StressSpec

Requirements Stress Tester

A user-friendly web application for analyzing requirement documents and detecting potential risks before development begins.

What is StressSpec?

StressSpec Overview

StressSpec acts as a **"wind tunnel" for requirements**, helping project managers, business analysts, and development teams identify hidden risks in requirement documents early in the development process.

Key Benefit: Detect problems before coding begins, saving time and money.

Quick Start

What You Need

- **Python 3.8 or higher** (check with `python --version`)
- **Internet connection** (for initial setup)

Getting Started - Step 1

Install Dependencies

```
pip install -r requirements.txt
```

This installs all required Python packages for StressSpec.

Getting Started - Step 2

Start the Web Application

```
python web_utils/run_web.py
```

Wait for the message: "**Application startup complete**"

Getting Started - Step 3

Open Your Browser

1. Navigate to: **http://127.0.0.1:8000**
2. You're ready to analyze requirements!

User Guide

Using the Web Interface

Step 1: Upload Your Requirements File

1. Open the web interface at http://127.0.0.1:8000
2. Click "**Choose File**" or drag and drop your requirements file

Supported formats:

- **.txt files** - One requirement per line
- **.md files** - Requirements in bullet points or numbered lists

Step 2: Analyze Your Requirements

After uploading, click **"Analyze Requirements"**

The system will automatically:

- Parse your requirements and assign unique IDs (R001, R002, etc.)
- Detect risks across 8 categories
- Calculate risk scores
- Identify the top 5 riskiest requirements

Step 3: Review Results

The results page shows:

- **Executive Summary** - Overview of detected risks
- **Top 5 Riskiest Requirements** - Prioritized list of requirements needing attention
- **Detailed Risk Breakdown** - All detected risks organized by category

Risk Categories Detected

1. **Ambiguity** - Vague or unclear language
2. **Missing Detail** - Incomplete specifications
3. **Security** - Missing security requirements
4. **Conflict** - Duplicate or contradictory requirements

Risk Categories Detected (Continued)

5. **Performance** - Missing performance specifications
6. **Availability** - Missing uptime/reliability requirements
7. **Traceability** - Missing requirement IDs or test references
8. **Scope** - Scope creep or boundary violations

Step 4: Download Reports

Download your analysis results in multiple formats:

- **HTML Report** - Professional, standalone report for stakeholder presentations
- **Markdown Report** - Detailed technical documentation
- **CSV Report** - Spreadsheet-compatible data for analysis
- **JSON Report** - Machine-readable format for integration

Understanding Risk Scores

How Risk Scoring Works

Each requirement receives a **Total Risk Score** based on detected issues:

- **Low** = 1 point
- **Medium** = 2 points
- **High** = 3 points
- **Critical** = 4 points
- **Blocker** = 5 points

Risk Score Example

Example:

- Requirement R001 has 2 risks: HIGH (3) + MEDIUM (2) = **Total Score: 5**
- Requirement R002 has 1 risk: CRITICAL (4) = **Total Score: 4**
- **R001 would be ranked higher** (more risky) than R002

Top 5 Riskiest Requirements

The system automatically identifies and highlights the 5 requirements with the highest risk scores.

This helps you:

- **Prioritize Review** - Focus on the most critical requirements first
- **Manage Risk** - Identify which requirements need immediate attention
- **Allocate Resources** - Allocate more time to high-risk items

Supported File Formats

Text Files (.txt)

One requirement per line:

```
The system shall allow users to login with email and password
The system shall display user dashboard after successful login
The system shall support password reset functionality
```

Markdown Files (.md)

Requirements in bullet points or numbered lists:

```
- The system shall allow users to login with email and password
- The system shall display user dashboard after successful login
- The system shall support password reset functionality
```

Note: Lines starting with `#` or `//` are automatically ignored as comments.

Setup & Installation

Automated Setup (Recommended)

Run the setup script to automatically configure everything:

```
python web_utils/setup_web.py
```

Automated Setup - What It Does

This will:

- Install all required dependencies
- Create necessary directories
- Download required libraries
- Verify the setup

Manual Setup

If you prefer manual setup:

Step 1: Install Dependencies

```
pip install -r requirements.txt
```

Manual Setup (Continued)

Step 2: Create Directories

```
mkdir -p uploads logs
```

Step 3: Start the Server

```
python web_utils/run_web.py
```

Alternative Ways to Start the Server

Method 1: Development Server (Recommended)

```
python web_utils/run_web.py
```

Method 2: Direct FastAPI

```
python -m uvicorn web.main:app --host 127.0.0.1 --port 8000 --reload
```

Alternative Ways to Start (Continued)

Method 3: Using Main App

```
python web/main.py
```

Accessing the Application

Once the server is running, you can access:

- **Main Web Interface**: http://127.0.0.1:8000
- **API Documentation**: http://127.0.0.1:8000/api/docs (for developers)
- **Health Check**: http://127.0.0.1:8000/health

Command Line Interface

(Advanced)

CLI - Basic Usage

For users who prefer command-line tools or want to integrate StressSpec into automated workflows:

```
# Analyze a requirements file and generate a Markdown report
python main.py --file data/sample_requirements.txt --report-format md --verbose
```

CLI - Different Report Formats

```
# Generate CSV report
python main.py --file requirements.txt --report-format csv --output report.csv

# Generate JSON report
python main.py --file requirements.txt --report-format json --output report.json

# Generate HTML report
python main.py --file requirements.txt --report-format html --output report.html
```

CLI Options

- `--file` - Path to your requirements file (required)
- `--report-format` - Output format: `md`, `csv`, `json`, or `html` (default: `md`)
- `--output` - Custom output file path (optional)
- `--verbose` - Show detailed processing information

Troubleshooting

Common Issue #1: Port Already in Use

Problem: The default port 8000 is already in use

Solution: Change the port in `web_utils/run_web.py` or use a different port:

```
python -m uvicorn web.main:app --host 127.0.0.1 --port 8001
```

Common Issue #2: Dependencies Not Installed

Problem: Missing required Python packages

Solution: Reinstall dependencies:

```
pip install -r requirements.txt --force-reinstall
```

Common Issue #3: File Upload Errors

Problem: File is too large or wrong format

Solution:

- Maximum file size: 10MB
- Supported formats: `.txt` and `.md` only
- Check that your file has proper line breaks

Common Issue #4: Module Import Errors

Problem: Python can't find the StressSpec modules

Solution: Make sure you're in the StressSpec project directory:

```
cd StressSpec
python web_utils/run_web.py
```

Getting Help

- Check the logs in the `logs/` directory for detailed error messages
- Verify all dependencies are installed: `pip list`
- Test the setup: `python -c "import fastapi, uvicorn, jinja2"`

Features

Core Capabilities

- **Automatic ID Assignment** - Each requirement gets a unique ID (R001, R002, etc.)
- **Line Number Tracking** - Maintains traceability to original file location
- **8-Category Risk Detection** - Comprehensive risk analysis across multiple dimensions

Core Capabilities (Continued)

- **Risk Scoring** - Combined risk scores per requirement
- **Top 5 Riskiest Requirements** - Automatic prioritization of critical issues
- **Multi-format Reporting** - Download reports in Markdown, CSV, JSON, or HTML
- **Real-time Analysis** - Fast processing with immediate results
- **Interactive Interface** - User-friendly web UI with responsive design

Risk Detection Categories

1. **Ambiguity** - Detects vague or unclear language
2. **Missing Detail** - Identifies incomplete requirements
3. **Security** - Flags missing security requirements
4. **Conflict** - Finds duplicate or contradictory requirements

Risk Detection Categories (Continued)

5. **Performance** - Identifies missing performance specifications
6. **Availability** - Detects missing uptime/reliability requirements
7. **Traceability** - Validates requirement IDs and test coverage
8. **Scope** - Flags scope creep and boundary violations

Additional Resources

Additional Resources

- **Project Documentation**: See `docs/StressSpec_Project_Progress.md` for detailed technical documentation, project planning, and development history
- **Sample Files**: Download sample requirement files from the web interface to test the application
- **API Documentation**: Available at `/api/docs` when the server is running (for developers)

Requirements

- **Python 3.8 or higher**
- **All dependencies** (installed via `pip install -r requirements.txt`)

Example Workflow

Example Workflow - Step 1

Prepare Your Requirements File

- Create a `.txt` or `.md` file with your requirements
- One requirement per line (or bullet point)

Example Workflow - Step 2

Start the Web Application

```
python web_utils/run_web.py
```

Wait for the server to start, then open your browser.

Example Workflow - Step 3

Upload and Analyze

- Open http://127.0.0.1:8000 in your browser
- Upload your requirements file
- Click "Analyze Requirements"

Example Workflow - Step 4

Review Results

- Check the Top 5 Riskiest Requirements
- Review detailed risk breakdown
- Filter by severity or category

Example Workflow - Step 5

Download Reports

- Choose your preferred format (HTML, Markdown, CSV, or JSON)
- Download and share with your team

Tips for Best Results

Tips for Best Results

- **Be Specific** - Clear, specific requirements result in fewer false positives
- **Use Consistent Format** - One requirement per line makes parsing more accurate
- **Review Top 5 First** - Focus on the highest-risk requirements first

Tips for Best Results (Continued)

- **Use HTML Reports** - Best for stakeholder presentations
- **Export to CSV** - Use for spreadsheet analysis and sorting

Need Help?

Getting Support

Need help?

- Check the troubleshooting section above
- Review the project documentation in `docs/StressSpec_Project_Progress.md`
- Check logs in the `logs/` directory for error details

Thank You

StressSpec - Requirements Stress Tester

For analyzing requirement documents and detecting potential risks