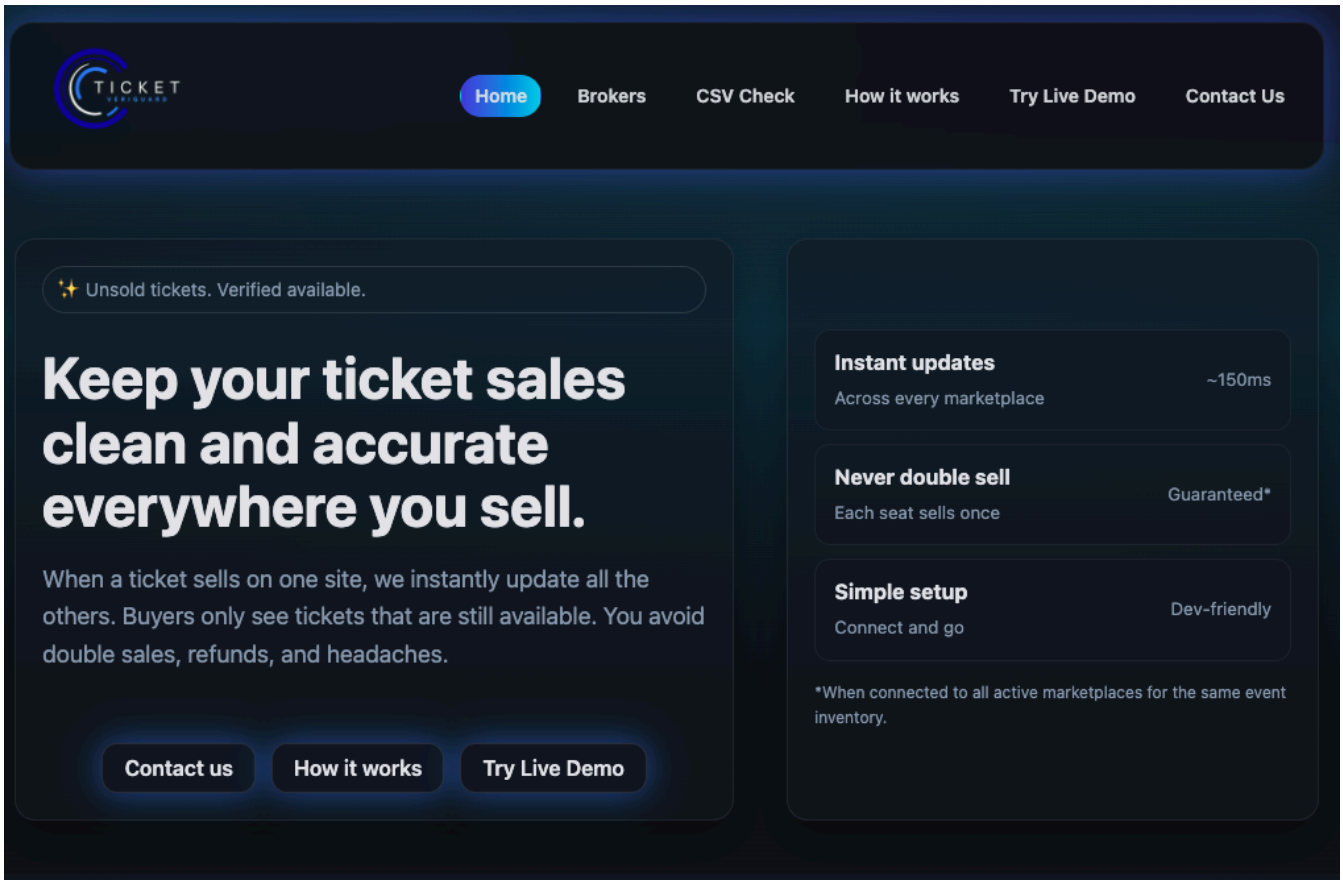
TicketVeriguard Case Study



Executive Summary

A responsive web prototype that helps ticket marketplaces prevent duplicate sales across multiple platforms by validating seat listings in real time.

Introduction

Ticket VeriGuard is a web-based solution designed to help brokers, resellers, and marketplaces automatically detect duplicate ticket listings before a sale occurs. I built this project to explore how smart data validation could improve trust between buyers and sellers while eliminating costly refund and reputation issues.

The system includes both a **marketing landing page** and a **functional demo app** that simulates how Ticket VeriGuard could operate across marketplaces using realistic seat and row data.

The Problem

Secondary ticket marketplaces often have sellers listing the same seats across multiple platforms. When one of those seats sells, the other listings remain active, causing **duplicate sales** and **refund disputes**.
These errors damage customer trust and strain relationships between sellers, marketplaces, and venues. Current tools rely heavily on manual updates, which are prone to error and delay.

I wanted to design a system that:

- Detects and blocks duplicate listings automatically
- Communicates results clearly to users
- Demonstrates how the concept could scale across marketplaces

My Role

I designed and developed both the marketing site and the live simulation prototype.
My responsibilities included:

- UI/UX design and front-end development
- API integration and data validation logic
- Messaging and branding for the product concept
- Deployment & testing on Render and GitHub Pages

Timeline

This project evolved through several phases over approximately **4 weeks**:

1. Initial concept and branding for Ticket VeriGuard
2. Landing page design and responsive layout
3. Functional prototype using real data simulation
4. Iterative testing and UX refinement

Tools and Skills Used

Tools and Libraries:

- HTML, CSS, JavaScript
- PapaParse (for CSV handling)
- Render (for backend hosting)

Backend Integration:

- FastAPI (Python) for validation logic
- Mock APIs simulating multi-marketplace traffic

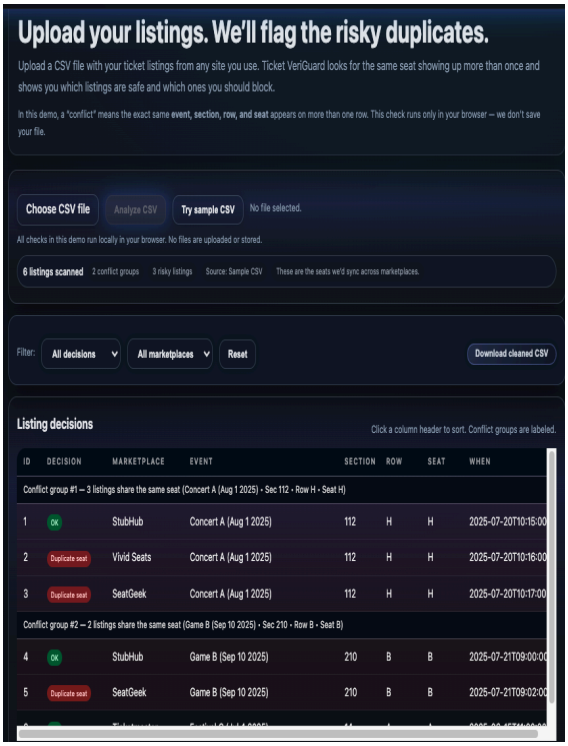
Skills I practiced:

- API integration and testing
- UI/UX design and prototyping
- Data validation and error handling
- Responsive design and deployment

Key findings

Key Finding 1:

Brokers and resellers want **instant visibility** into which listings conflict



before they post tickets to multiple sites.

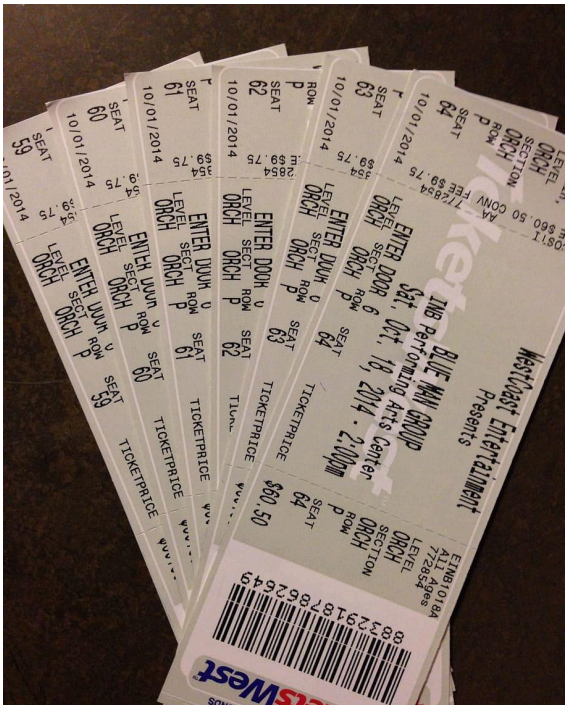
Key Finding 2:

Real-time validation creates trust and users felt more confident listing their tickets when they could see results immediately.

Key Finding 3:

Simple dashboards with clear color coding (green for approved, red for blocked) helped non-technical users understand complex seat data at a glance.

Solutions



Solution 1

Used seat and row parsing to detect duplicate listings even when formatting varied (e.g., “Row A Seat 3–4” vs “A3, A4”).

Solution 2

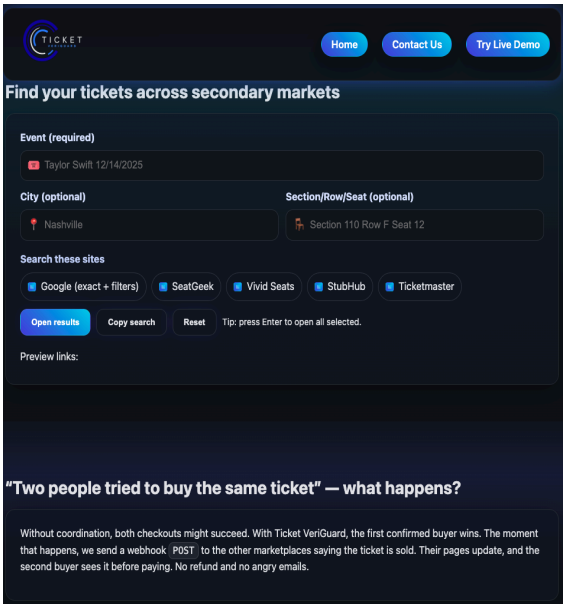
Built real-time feedback into the prototype, allowing users to see validation results as they typed or uploaded CSV data.

Solution 3

Added filtering options for marketplace view, so users could toggle between their listings and the full marketplace dataset.

Solution 4

Developed a landing page that communicates the concept clearly and guides visitors toward trying the demo.



Impact

Impact 1

Demonstrated how automation can eliminate duplicate ticket sales, reducing refund costs and boosting trust between brokers and marketplaces.

Impact 2

Provided a clear, visual example of how data validation can scale across marketplaces.

Impact 3

Allowed me to expand my skills in full-stack development, API design, and communicating complex data in a user-friendly way.

Case Study Introduction

Background

This project was part of my independent development portfolio, designed to combine technical experimentation with real-world relevance.

I wanted to apply my web development training toward a practical, industry-level problem — something that would resonate with both users and potential business partners.

Objectives

Objective 1

Design a proof-of-concept system for detecting duplicate seat listings across marketplaces.

Objective 2

Build a simple, intuitive dashboard where users can test the process.

Objective 3

Create a companion landing page that introduces the concept and potential value.

Objective 4

Communicate technical accuracy through plain language and clean design.

Problem

Duplicate ticket sales harm buyers and sellers alike, leading to lost revenue, bad publicity, and customer frustration.

Marketplaces need a way to **check listings across platforms instantly** — not after a sale. Ticket VeriGuard solves this with

smart data validation logic and a clean, educational user interface.

Recommendations

Recommendation 1

Connect marketplaces through a shared validation endpoint to prevent duplicate listings.

Recommendation 2

Give brokers insights into which venues or events generate the most conflicts.

Recommendation 3

Allow partners to test validation integrations safely before rollout.

Impact of recommendation

SOLUTION	EFFECTIVENESS	IMPACT	NOTES
Real-time seat validation	Highly effective	Prevented duplicate listings and errors	Core concept of the demo
CSV upload and feedback dashboard	Highly effective	Made data validation accessible and visual	Helped non-developers understand results
Marketplace filters	Effective	Let users compare results between sources	Added practical flexibility
Landing page communication	Highly effective	Helped pitch concept to non-technical audiences	Supported business storytelling

Analysi

Research methods

- Reviewed common duplicate-ticket scenarios on secondary markets
- Conducted user feedback sessions with sellers familiar with StubHub and Vivid Seats
- Tested the prototype’s behavior on Chrome, Firefox, and Safari

Approaches used

- Client-side validation with JavaScript
- Backend API calls for conflict detection
- Responsive grid layout using CSS Flexbox
- Deployment using GitHub Pages + Render

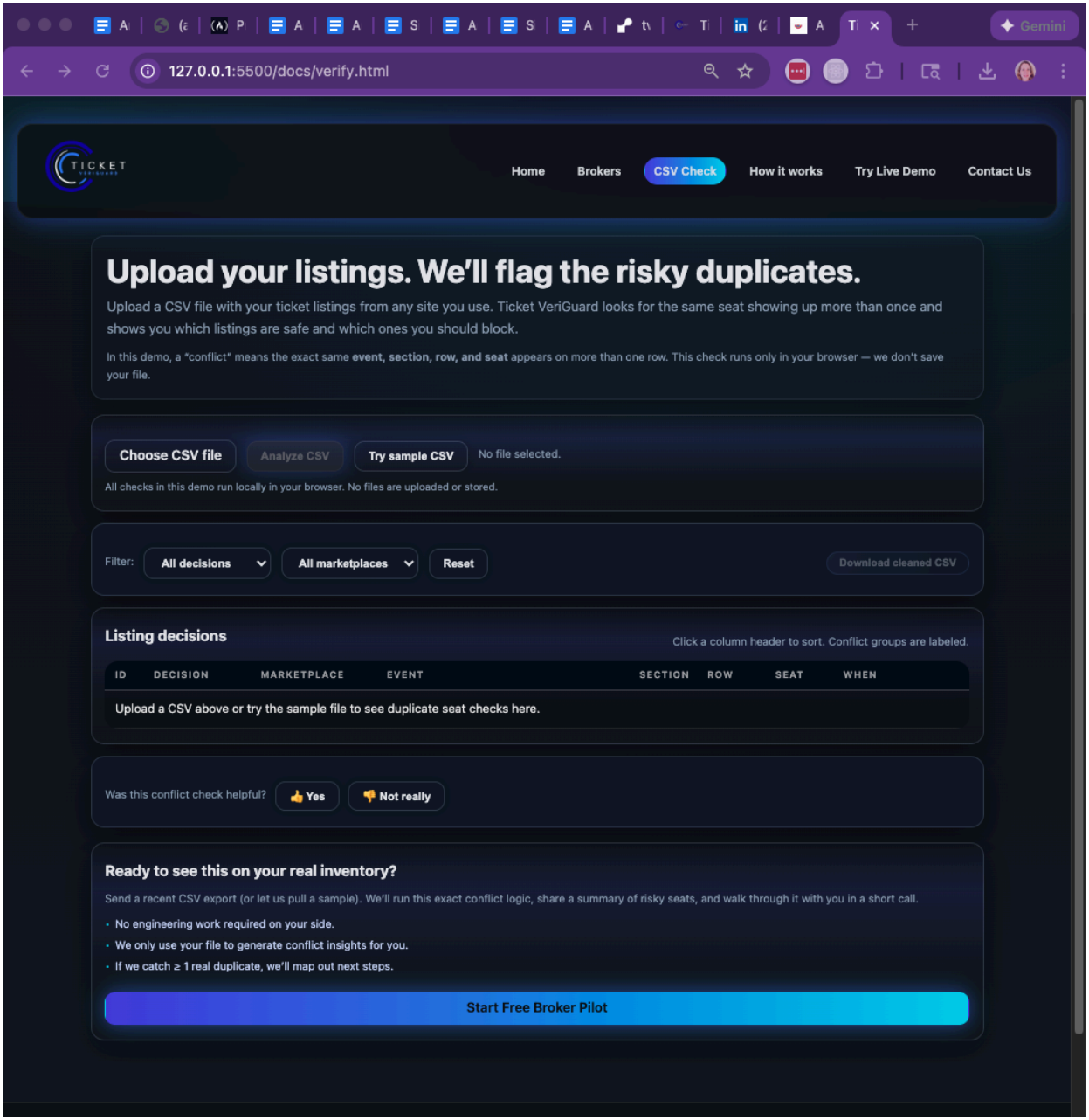
Relevant facts and information

- Duration: 4 weeks
- Tools: HTML, CSS, JavaScript, FastAPI
- Features: Live validation, CSV upload, marketplace filtering, branded landing page

Challenges

- Ensuring seat range parsing worked across inconsistent user input
- Balancing speed with accuracy in validation logic
- Creating a consistent brand story for both technical and non-technical audiences

Conclusion



Summary of findings

Ticket VeriGuard demonstrates how web technologies can prevent duplicate ticket sales through real-time validation and intelligent data parsing.

The system simplifies a major marketplace pain point and shows how automation can make secondary ticket sales more trustworthy.

Implications of the study

This project gave me deeper experience in full-stack problem-solving, user experience design, and communicating technical solutions for real business challenges.

It also laid the groundwork for future development and potential real-world application with ticket marketplaces.

> **“Ticket VeriGuard taught me how to bridge technical systems and business storytelling — creating something that’s both smart and simple enough for everyone to use.”**