Automated Transfer Credit Evaluator

S2 Project Plan

Tyler Dionne, Kendall Kelly

Motivation

Objective: Simplify transfer credit assessment for Florida Tech students by providing:

- Instant online transcript evaluation
- Immediate, accurate credit acceptance reports
- Reduced uncertainty in transfer process

Key Benefits:

- User-friendly digital platform
- Eliminates manual credit evaluation delays
- Empowers students to make informed transfer decisions quickly

Key Features:
User System Architecture

Two-Tier User Access:

- User Level:
  - Access tool and upload transcript
  - Receive credit evaluation
  - Create user level account
- Admin Level:
  - Backend management
  - site analytics

Identity and Access Management (IAM):

- Separate resource access
- Prevent privilege escalation

Key Features: Two User Inputs

Transcript Upload

- Supports .txt and .pdf formats
- Contains:
    - Information on previously taken courses
    - Number of credits for each course
    - A brief course description

Catalog Selection

- Choose Florida Tech degree program
- Determines credit transfer evaluation

Key Features: Evaluation and Technology

Detailed Transfer Credit Results:

- Comprehensive table display
- Shows:
    - Accepted credits
    - Course numbers
    - Number of credits for each course
    - Equivalent Florida Tech courses

Full Stack Web Application

- Built with Flask
- Integrated database
- Complete standalone platform

Algorithms & Tools

Front-end

- Languages: HTML, CSS, JavaScript
- Hosting: GitHub Pages
- PDF Parsing: PDF.js

Back-end

- Framework: Flask (Python)
- Database: SQLite3
- Supports:
  - Catalog storage
  - User login information
  - Web application integration

Novel Features

Automated Evaluation

- Eliminate manual credit input
- Simple file upload process
- Instant transfer credit comparison

Student Empowerment

- Independent credit evaluation
- Transparent transfer process
- Simplified institutional transfer

Enhanced Security

- Two-tier user system
- Admin-only access to sensitive information
- Protect system integrity

Technical Challenges: Course Matching Algorithm

We need to develop an algorithm for course equivalency.

Match courses using:

- Transcript details
- FIT course descriptions

Credit acceptance will be determined based on similarity.

Technical Challenges: Implementing Security

- Two-Tier User System
  - Restrict user access
  - Prevent unauthorized server/data access
- Admin-only privileges for:
  - Source code
  - Usage analytics
  - Backend resources

Technical Challenges: Web App Security

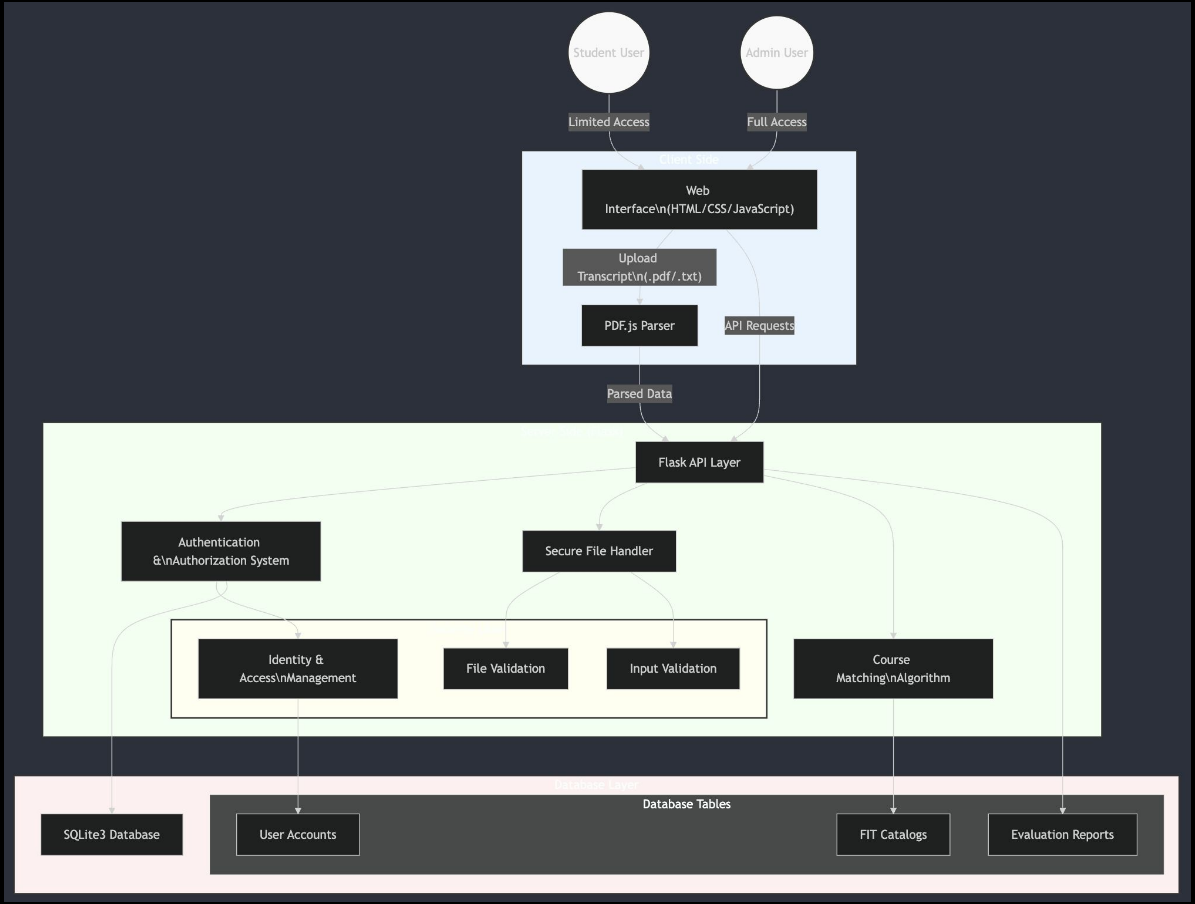
Mitigate Web Vulnerabilities

- Use Flasks built in protections to ensure the website can handle common vulnerabilities
- Conduct penetration testing once the website is complete

Secure File Handling

- Validate file uploads
- Prevent malicious script execution
- Restrict file types beyond extensions

System Architecture Diagram



Progress

Completed

- Website Frontend (HTML/CSS)
- Text File Processing
- PDF File Processing

To Do:

- Flask Full Stack Conversion
- User/Admin Account System
- Web Application Security
- Secure File Upload Mechanisms
- Basic Pentesting

Milestone 4: Key Tasks

Flask Setup

- Virtual environment configuration
- Install required components
- Develop application structure

Frontend Migration

- Convert HTML to Flask templates
- Update CSS/JavaScript handling
- Verify page display

Database Integration

- Design SQLite schema
- Create migration scripts
- Implement CRUD operations
- Manage course catalogs, credit evaluations

File Upload System

- Secure file handling (txt/pdf)
- Safe file storage

API Development

- Manage transcript uploads
- Handle catalog selection
- Provide evaluation results
- Implement error handling

Questions?