

Team Members:

Tyler Dionne (tdionne2021@my.fit.edu), Kendall Kelly (kelly2021@my.fit.edu), Braden Corkum (corkumb2013@my.fit.edu)

Project Advisor:

Sneha Sudhakaran, ssudhakaran@fit.edu

Project Title:

FIT Automated Transfer Credit Evaluation

Client:

Sneha Sudhakaran

Website:

<https://tylerdionne.github.io/ATCE-FIT/index.html>

Milestone 3 Progress Evaluation

1. Progress of Current Milestone:

Task	Completion %	Tyler	Kendall	Braden	To Do
Transcript creation.	100%	33.3 %	33.3%	33.3%	Begin to modify the complexity of data and extensions supported incrementally as progress is made on the final product.
ATCE Tool User Interface Improvements: Drag & Drop, Visual Appeal, User-friendly.	100%	50%	50%	0%	Implement storage of FIT catalogs in a DB once the backend is implemented.
PDF preprocessing and generating a format for parsing to front end.	100%	0%	0%	100%	Implement techniques to identify what data to take from transcripts.
PDF Parsing Logic: integrated with the	100%	50%	50%	0%	Implement backend functionality with sql lite

<p>html front end for the atce.html webpage. Improvements: txt, and pdf support, html scripts to parse data from a pdf, verify format constraints, and display the information in a table to the user.</p>				<p>DB to store catalogs, logins. Use python Flask to implement backend functionality to store and retrieve data and run the entire web app independently with docker. Improve pdf parsing with more than one line per file (in progress) and a more realistic schema for the format expected in the transcript.</p>
--	--	--	--	---

2. Discussion of Each Completed Task:

Transcript creation

Braden Corkumb - obtained 15 college university transcripts online in a .pdf format.

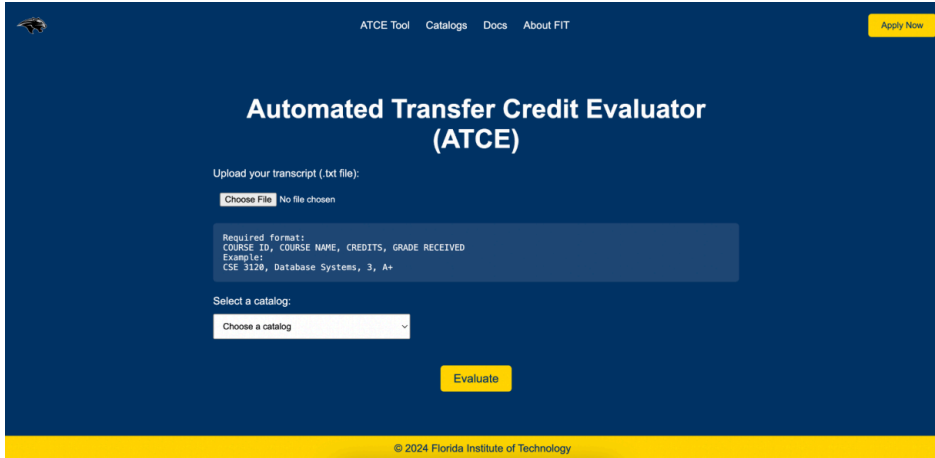
Kendall Kelly - obtained 10 transcripts

Tyler Dionne - obtained 10 transcripts. created 10 transcripts in .pdf format to test newly implemented pdf parsing features in the atce tool.

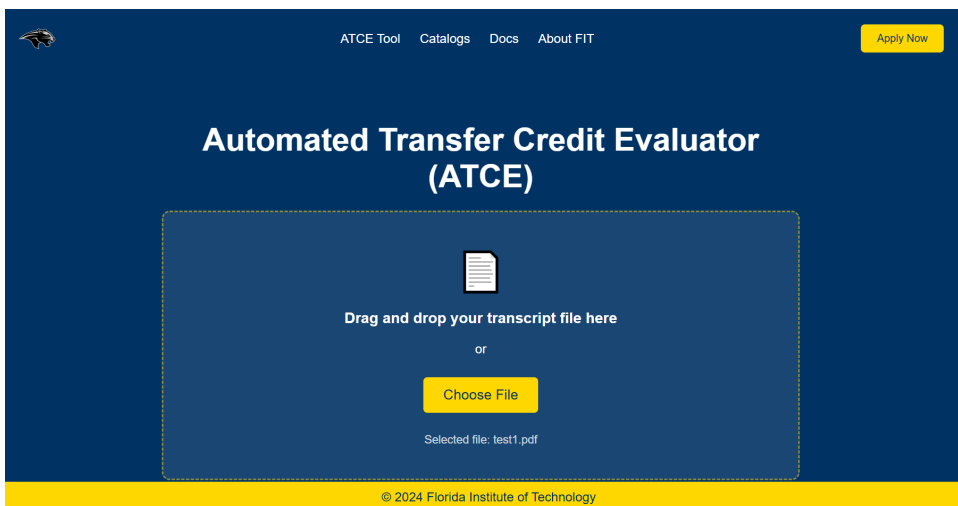
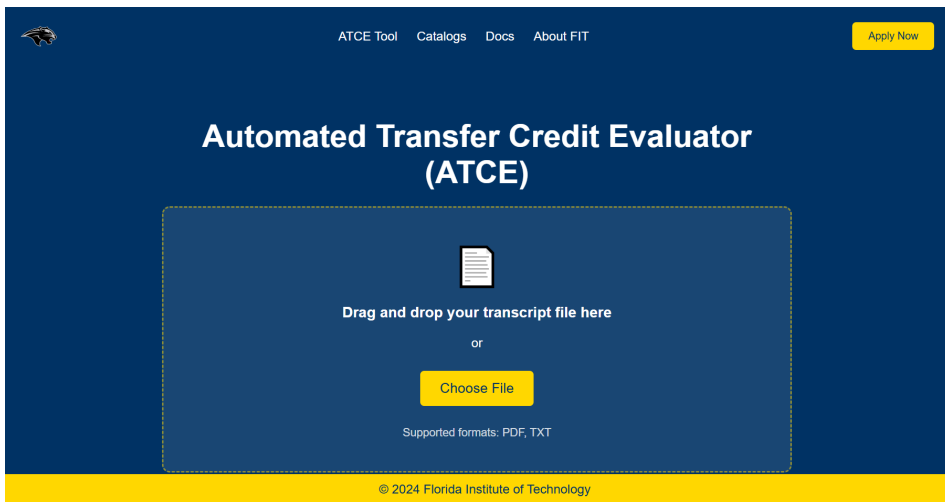
ATCE Tool User Interface Improvements: Drag & Drop, Visual Appeal, User-friendly PDF Parsing Logic: Integrated .pdf support with the html front end for the atce.html webpage. Improvements: pdf support, html scripts to parse data from a pdf & verify format constraints, display the information to the console.

ATCE Tool User Interface Improvements

Previous:



Updated:



1. Visual Appeal

Previous: Visually un-appealing to the user. Does not quite look like a tool. Not convenient and or easy to use. Outdated UI design.

Updated: The visual appeal of the newly designed UI is a great improvement given the use of lighter colors in the backdrop and icons in contrast to the outdated, dull design initially. Visually looks more like a tool that you would find online in 2024 in the updated version rather than ancient math design previously.

2. Drag & Drop Functionality:

Previous: Visually un-appealing to the user. Does not quite look like a tool. Not convenient and or easy to use. Outdated UI design.

Updated: The Drag & Drop Functionality improves the usability of the tool and makes for an improved user experience by providing the option to choose whether to select a file traditionally or drag and drop. Drag and Drop features are popular among users who may want to upload various files at one time.

3. User-friendly

Previous: Visually un-appealing to the user. Does not quite look like a tool. Not convenient and or easy to use. Outdated UI design.

Updated: With the improved UI design, convenient file uploading and extended support for pdf files the tool and front end of the web application as a whole has had an increase in user friendliness, practicality and usability.

1. *.pdf Support*

Scripts were integrated directly into the front end html for the atce tool. Thus far the tool can read in .txt files and .pdf files in a specific format.

The tool utilizes the PDF.js library to handle PDF file processing. This library is initialized at the beginning of the script:

```
pdfjsLib.GlobalWorkerOptions.workerSrc =  
'https://cdnjs.cloudflare.com/ajax/libs/pdf.js/3.11.174/pdf.worker.m  
in.js';
```

When the user uploads the pdf, the processPDF function is called and deals with extracting text content from the file.

```
async function processPDF(file) {
  try {
    const arrayBuffer = await file.arrayBuffer();
    const pdf = await pdfjsLib.getDocument({ data: arrayBuffer
  }).promise;
    let extractedText = '';
    progress.style.width = '30%';

    for (let i = 1; i <= pdf.numPages; i++) {
      const page = await pdf.getPage(i);
      const textContent = await page.getTextContent();
      const pageText = textContent.items.map(item =>
item.str).join(' ');
      extractedText += pageText + '\n';
      progress.style.width = `${30 + (70 * i /
pdf.numPages)}%`;
    }

    const lines = extractedText.split('\n');
    validateAndProcessContent(lines);
  } catch (error) {
    formatError.textContent = 'Error processing PDF: ' +
error.message;
    formatError.style.display = 'block';
    progressBar.style.display = 'none';
  }
}
```

The function extracts the text by:

- Converting the file to an ArrayBuffer
- Using the PDF.js library to load the document and then iterate through each page of the pdf and extract the text content
- Updating a progress bar to the user
- Splitting the text from the .pdf file into lines and passing them to validateAndProcessContent

After the text is extracted and sent to the `validateAndProcessContent` function it makes sure that the content follows the desired format and if so it is validated.

```
function validateAndProcessContent(lines) {
  let isValid = true;
  const processedLines = [];

  for (let line of lines) {
    line = line.trim();
    if (line === '') continue;
    const parts = line.split(',').map(part => part.trim());
    if (parts.length !== 4) {
      isValid = false;
      break;
    }
    if (isNaN(parts[2])) {
      isValid = false;
      break;
    }
    if (!parts[3].match(/^[A-F][+-]?$/)) {
      isValid = false;
      break;
    }
    processedLines.push({
      courseID: parts[0],
      courseName: parts[1],
      credits: parseFloat(parts[2]),
      grade: parts[3]
    });
  }

  // ... (code for handling validation results)
}
```

The function checks each line to make sure it contains four comma-separated values (course ID, course name, credits, and grade) and it verifies that the credits are numeric and grade follows the expected format (A+). If the content passes the checks it makes an array of processed course objects and the `displayResults` function is called to present the data to the user/console.

```

function displayResults(processedLines) {
  const resultsContainer =
document.getElementById('results-container');
  const resultsBody = document.getElementById('results-body');
  const totalCreditsCell = document.getElementById('total-credits');

  resultsBody.innerHTML = '';

  const totalCredits = processedLines.reduce((sum, course) => sum +
course.credits, 0);

  processedLines.forEach(course => {
    const row = document.createElement('tr');
    row.innerHTML = `
      <td>${course.courseID}</td>
      <td>${course.courseName}</td>
      <td>${course.credits}</td>
      <td>${course.grade}</td>
    `;
    resultsBody.appendChild(row);
  });

  totalCreditsCell.textContent = totalCredits.toFixed(1);
  resultsContainer.style.display = 'block';
}

```

The function creates a table row for each course, inserts the extracted information, and calculates the total credits. The results are then displayed in a table directly below the atce tool.

Automated Transfer Credit Evaluator (ATCE)



Drag and drop your transcript file here

or

Choose File

Selected file: test1.pdf

Expected format in document:
COURSE ID, COURSE NAME, CREDITS, GRADE RECEIVED
Example:
CSE 3120, Database Systems, 3, A+

Transcript Analysis Results

Course ID	Course Name	Credits	Grade
CSE 3120	Database Systems	3	A+
Total Credits:		3.0	

By implementing support for .pdfs, we have improved the atce tool to become more practical and user friendly.


Testing

To test the new implementation of .pdf support, the team created a small data set of example transcripts in .pdf format to ensure the script is validating the requirements properly.

test-1.pdf

CSE 3120, Database Systems, 3, A+

Automated Transfer Credit Evaluator (ATCE)



Drag and drop your transcript file here

or

Choose File

Selected file: test-1.pdf

Expected format in document:
 COURSE ID, COURSE NAME, CREDITS, GRADE RECEIVED
 Example:
 CSE 3120, Database Systems, 3, A+

Transcript Analysis Results


Course ID	Course Name	Credits	Grade
CSE 3120	Database Systems	3	A+
Total Credits:		3.0	

Default test case.

test-4.pdf

CSE 1002 , Intro To Programming, , B+

Automated Transfer Credit Evaluator (ATCE)



Drag and drop your transcript file here

or

Choose File

Selected file: test-4.pdf

Expected format in document:
 COURSE ID, COURSE NAME, CREDITS, GRADE RECEIVED
 Example:
 CSE 3120, Database Systems, 3, A+

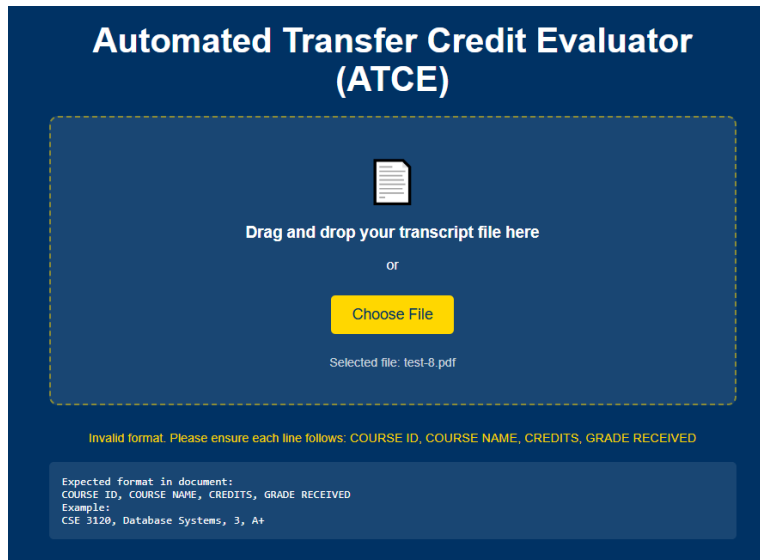
Transcript Analysis Results

Course ID	Course Name	Credits	Grade
CSE 1002	Intro To Programming	NaN	B+
Total Credits:		NaN	

If no credits are specified we see there is no error printed but instead the NaN (Not a Number) value is placed in the credits column. This test case help us to find a bug and understand what additional checks need to be done.

test-8.pdf

CSE 1002 , Intro To Programming, 2 , G



We see here that the script picks up on any letter outside of the range A+ to F- and hence when we have G as a grade we have invalid format.

By developing tests for the new implementation we can use edge cases and fuzzing techniques to find any unexpected functionality of our program. These test cases help us to understand bugs and additional checks that need to be done in order to validate the format correctly in each scenario.

Developing test cases helps the team to find bugs in our html script and understand additional checks that need to be done to validate format.

3. Team Member Contribution of Milestone 3:

Braden Corkumb - Obtained 15 college university transcripts online in a .pdf format. Will develop techniques to identify what data might be critical and how the transcript data can be used to match with transfer credits. Also, will learn more about the different transcripts and make sure preprocessing is done and this preprocessed information will later be integrated to ATCE tool.

Tyler Dionne - Obtained 10 transcripts and created 10 transcripts in .pdf format to test newly implemented pdf parsing features in the atce tool. ATCE Tool User Interface Improvements: Drag & Drop, Visual Appeal, User-friendly. PDF Parsing Logic: Integrated .pdf support with the html front end for the atce.html webpage. Improvements: pdf support, html scripts to parse data from a pdf & verify format constraints, display the information to the console.

Kendall Kelly - Created 10 transcripts in .pdf format to test newly implemented pdf parsing features in the atce tool. Worked on improving UI attributes such as general visual appeal and drag and drop features for files. Helped integrate PDF support with the html front end for the tool page. Helped write the html scripts to parse data from PDF files and verify the format constraints. Also assisted in displaying the information to the console.

4. Plan for Milestone 3:

Task	Tyler	Kendall	Braden
Given FERPA regulation, determine how to generate a corpus of transcripts for testing.	Tyler will work to find a solution to legal regulations with obtaining genuine university transcripts online. Research ways to generate a dataset of generated, realistic transcripts.	Kendall will work to find a solution to legal regulations with obtaining genuine university transcripts online. Research ways to generate a dataset of generated, realistic transcripts.	Braden will work to find a solution to legal regulations with obtaining genuine university transcripts online. Research ways to generate a dataset of generated, realistic transcripts.
Begin to modify the complexity of data and extensions supported incrementally as progress is made on the final product.	Tyler will find and recommend data extension complexities for ACTE tool.	Kendall will recommend data extension complexities for ACTE tool.	Braden will work to begin to modify the complexity of data and extensions supported incrementally as progress is made on the final product.
Implement storage of FIT catalogs in a DB once the backend is implemented.	Tyler will implement storage of FIT catalogs in a DB once the backend is implemented.	Kendall will work to implement storage of FIT catalogs in a DB once the backend is implemented.	Based on input of data to be fit in Braden will make sure the data given for querying databased is in standard format

Implement backend functionality with sql lite DB to store catalogs, logins. Use python Flask to implement backend functionality to store and retrieve data and run the entire web app independently with docker.	Tyler will work to implement backend functionality with sql lite DB to store catalogs, logins. Use python Flask to implement backend functionality to store and retrieve data and run the entire web app independently with docker.	Kendall will use sql lite DB to start implementing the backend. SQLite will store the catalogs and logins. Flask will be used to implement backend functionality to store and retrieve data and run the entire web app.	Braden will test the tool and update how the DB works
Improve pdf parsing with more than one line per file (in progress) and a more realistic schema for the format expected in the transcript.	Tyler will work to improve pdf parsing with more than one line per file (in progress) and a more realistic schema for the format expected in the transcript.	Kendall will try to improve pdf parsing with more than one line per file.	Braden will take over Pdf parsing and make a more realistic schema for the format expected in the transcript.

5. Date(s) of meeting(s) with Client during the current milestone:

- Once a week every two weeks

6. Client feedback on the current milestone:

- See Faculty Advisor Feedback below

7. Date(s) of meeting(s) with Faculty Advisor during the current milestone:

- Once a week every two weeks

8. Faculty Advisor feedback on each task for the current Milestone:


Faculty Advisor Signature: _____ Date: 11/22/2024

Evaluation by Faculty Advisor

Faculty Advisor: detach and return this page to Dr. Chan (HC 209) or email the scores to pkc@cs.fit.edu

Score (0-10) for each member: circle a score (or circle two adjacent scores for .25 or write down a real number between 0 and 10)

Tyler Dionne	0	1	2	3	4	5	5.5	6	6.5	7	7.5	8	8.5	9	9.5	10
Kendall Kelly	0	1	2	3	4	5	5.5	6	6.5	7	7.5	8	8.5	9	9.5	10
Braden Corkum	0	1	2	3	4	5	5.5	6	6.5	7	7.5	8	8.5	9	9.5	10

Faculty Advisor Signature: _____  _____ Date: 11/22/2024