

**User Manual**

**Senior Project**

Tech Tutor

**Version**

1.0

**Product Owner**

Mathew Brimberry

San Juan Unified School District

**Team Byte Brigade**

Angelo Karam

Joshua Grindstaff

Ryan Naveira

Austin Nolte

Jacob Sherer  
 Collin Dunkle

Joaquim Pedroza

Chisom Iwunze

**Table of Contents**

[Introduction 3](#_Toc968620102)

[Installing Program 3](#_Toc1734478013)

[Google Drive Integration 4](#_Toc1374069699)

[Adding Google Account Key 5](#_Toc1191056323)

[Google Sheet Selection: 6](#_Toc1849911284)

[The User Interface 6](#_Toc1071949553)

[What Happens Behind the Scenes 7](#_Toc211945368)

[File Upload Guidelines 9](#_Toc1317856253)

[System Requirements 9](#_Toc1276560131)

# **Introduction**

TechTutor is a project developed by the Byte Brigade team, designed to streamline the grading process for educators. This Python-based application utilizes PyTorch, a powerful machine learning library, to analyze images of students' homework submissions. The program simplifies the grading process by allowing teachers to input a grading key, which is then used to assess and score the submitted assignments accurately. Once graded, the results are automatically uploaded to Google Drive, providing a seamless workflow for educators to review and manage. By utilizing moden AI technology, Tech Tutor can enhance and educator’s consistency and reduce their workload to allow them to focus on more important tasks.

# **Installing Program**

**Prerequisites**

* Install Miniconda
* Ensure that graphics drivers are enabled and up to date

**Option 1: Manual Installation**

* Download Miniconda for your platform from Miniconda Downloads.
* Follow the installer instructions for your operating system.
* Verify installation by running:

conda --version

**Option 2: Automated Installation**

* After cloning the repository use the provided InstallMiniconda.bat script.
* Navigate to the directory containing the script.

InstallMiniconda.bat

* After the script completes, verify installation by running:

conda --version

**Step 1: Clone the Repository**

* Open a terminal and navigate to your desired folder.

git clone https://github.com/Very-Bad-Goose/Senior-Project.git

* Navigate into the project directory:

cd Senior-Project

**Step 2: Run the Deployment Script**

* Execute the provided deployment script:

run\_app.bat

**The script will:**

* Use the provided .yml file to create and configure the necessary Conda environment.
* Install all dependencies required for the project.

**Step 3: Run the Program**

* Inside the project directory, start the program by running:

python main.py

**Desk Image Orientation**

The images of the top of desk will be rotated counterclockwise. When taking the picture, the top of the photo should be on the left side of the desk. In other words, the top of the phone pointed to the left of the desk and the right of the phone is pointed to the right

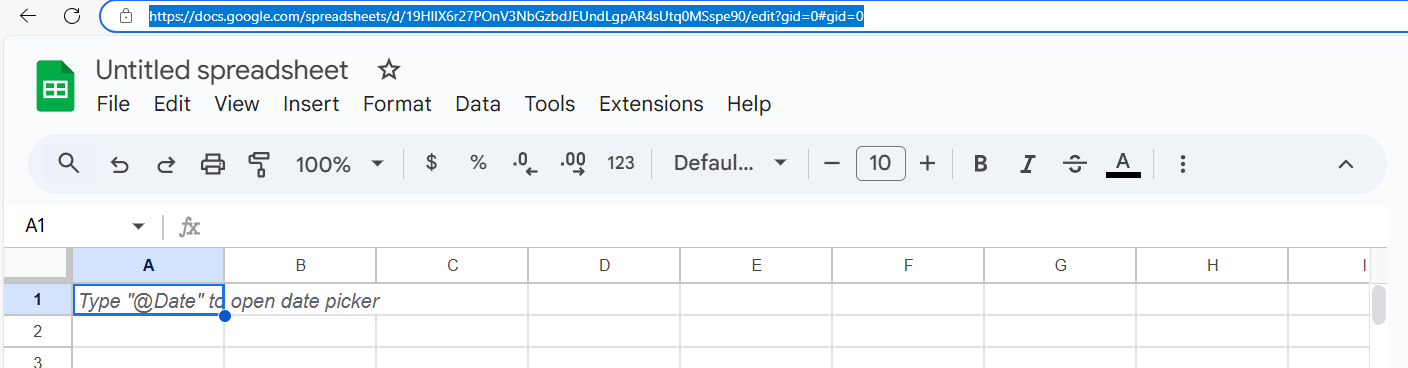
# **Google Drive Integration**

TechTutor automatically uploads results to a designated Google Drive folder.  
Steps to configure:

1. Log into Google drive with the account you would like to use with TechTutor, using any browser of your choosing.
2. Create a google cloud project: <https://console.developers.google.com/>. Further information on doing so can be found at: <https://support.google.com/a/answer/7378726?hl=en>
3. Add Google Sheets and Google Drive API to your project. This can be done through the APIs & Services page or, if a search bar is available, it can be searched directly.
4. Create a Service account. In the same page, navigate to the service accounts page which can again be found using the search bar. Once on the service account page click on the button which says to create a service account and give it a name and ID (description optional).
5. Add a service key to your service account. Once the service account has been created, click on the link to the service account that you just made on the service account page and navigate to the keys tab. Once there, click on the create key button and create a new key using the JSON format. The key will download itself onto your computer. You will need to save that key for later when we start using the program.
6. Copy the email address of the service account and share the Google Sheet with that account. In addition, share the folder of the Google Drive holding all of the images that you are using with that account as well so it has access to them.

# **Google Sheet Selection:**

The Google Sheets integration in TechTutor allows educators to seamlessly connect to and update their grading sheets. This feature uses the Google Sheets API to pull data like student IDs and update the grades after processing. Educators can input the Google Sheet ID into the application and save it for repeated use.



Enter the URL of the google sheets you want to grade the work in and ender it into the text box called “Enter Sheet ID” then click “Save Sheet ID” Button

**Key Features:**

Sheet ID Entry: The user enters the Google Sheet ID via a dedicated text box. This will need to be entered every time the application is run if you are running this for a new sheet. If not then the application will run for the last entered sheet and could overwrite existing information.  


Save Button: Saves the entered Sheet ID to a .txt file for future reference.  


Change Account Credentials: Uploads a JSON file containing the credentials required for Google Sheets API access. This will need to be run once initially upon application installation and then it will be stored for future uses.  


Packet Pages Field: The user enters the amount of pages/images that the activity packet folder should contain.  


Save Number: This button will save the number entered in the Packet Pages Field and store it in our application for comparisons. If the number is valid (greater than 0) then it will display a label saying “Number Saved”. If it is invalid it will display “Invalid Number”.



**Workflow:**

The user enters the Sheet ID.

The application validates and accesses the Google Sheet.

Grading data is updated row by row using AI-powered image analysis.

Results are displayed to the user in real-time.

# **The User Interface**

TechTutor has a simple UI that facilitates our client in starting or stopping the AI from carrying out its work. It also gives the client extra insight into what the AI is currently doing by showing the current document it is working with.

Buttons and their functions in logical order:

Sheet ID entry field: allows you to enter a Google Sheet ID, which is necessary for the API to connect the AI to the desired Google Sheet.

Save Sheet ID Button: saves the sheet ID name to a .txt file in the src directory

Change Account Credentials Button: allows you to select a .json file from your file explorer. The .json file contains the account credentials required to read/write to the specified Google Sheet (Google API credentials). This saves the credentials in a configuration file, used in the background while the AI is running.

Start Button: starts the evaluation process of the AI on the selected Google Sheet.

Stop Button: stops the evaluation process of the AI.

Progress Bar: a visual representation of the number of documents that have been evaluated versus the number of documents left to evaluate.

Packet Pages entry field: allows you to enter the number of expected images for a packet submission. If a student’s submission does not meet the amount of expected images then it will give them a 0 grade.

Save Number button: Saves the number entered in the packet pages entry field so that the application can use it in its comparisons.

# **What Happens Behind the Scenes**

Once the user runs the following command from the terminal in the project directory python main.py:

1. The program will begin loading the models, id\_periodNum\_model.pt, desk\_model.pt, and caddy\_model.pt. These models will be loaded for future use of predicting the necessary image crop locations to make this program work. The models should be in the /models folder.

When you click start

1. The program will start by parsing through your program's directory for a config.txt file. This file should contain the path to your google cloud Json file. This file should contain the contents pertaining to your google service\_account. Vital information that should not be shared should be in this Json file such as private\_key\_id. This file will allow the program to make changes to a google sheet that is linked.
2. The program will begin the main loop of pulling the folders out of specified spreadsheet locations, specifically column F. This column should contain hyperlinks to google drive folders for the program to row by row download locally and then delete after it is done with said folder.

Per folder

1. The application will start parsing through the files in the folder looking for PNG files. These png files will be passed into their respective model based on name of file. The names are activity\_x.png and desk\_1,.png desk\_2.png where x is a number for the amount of pictures. The activity images will be passed into the id\_periodNum\_model.pt where the model will look for student ID and period number boxes. These boxes will be cropped out of the picture and passed into the handwriting\_recognition\_model.pt. The handwriting model will try and read the handwriting of the student and determine the student ID and period number written. These numbers will be compared to the current row the application is working on and assign a 0 if the student ID and period number do not match the correct values given in the sheet.
2. The other images get passed into the other two models. It is assumed that desk\_1.png is the top of the desk so therefore it will be passed into desk\_model.pt. This model will look for calculators and desk numbers and then be read by easyOCR to determine the numbers. Again like the activities, if the number and calculator number do not match the correct values corresponding to the sheet for that student they will be assigned a 0.
3. Lastly desk\_2.png is assumed to be the caddy of the desk. Where it is sent to caddy\_model.pt where it looks for desk numbers to be read by easyOCR. This process is much like desk\_1.png, just with a new model that is more specialized for desk caddy look up.

While the models are working

1. You will see the progress bar move depending upon how many rows are left to go. It will update once it is done with a row.

**Troubleshooting**

If the program crashes when clicking start

1. Ensure that a service account if selected by following the instructions for Adding Google Account Key.
2. Ensure that the google sheet ID entered is pointed to the correct google sheets and that the google API account has edit permissions for the google sheet.

If the program does not start

1. Download OpenGL 2 if a pop up comes up telling you to download it
2. Ensure the correct models are in the models folder there should be a caddy\_model.pt, desk\_model.pt, handwriting\_recognition\_model.pt, and an id\_periodNum\_model.pt. These models should be bigger then 1kb. If they are exactly one kb then you must request access for the models from one of the team members

# **File Upload Guidelines**

1. The program currently supports PNG files only. Images in other formats (e.g., JPEG, PDF) must be converted to PNG before submission.
2. To ensure accurate processing, uploaded images should be clear and legible. Avoid extremely low-resolution submissions.
3. Images must be well-lit, without excessive shadows or glare.
4. Blurry images will not be processed and will be flagged.

# **System Requirements**

At least 22GB of storage