Work done:

1) **Naming Convention Enforcing** (Python)

   Repo link: People-Places-Solutions/Enforcing-Naming-Convention: Convert into Jacobs standard naming convention (github.com)

   - BERT model building
     - Transformer model – encoder, decoder structure
     - This code uses BERT-Base model - Enforcing-Naming-Convention/BERT_Model.py at main · People-Places-Solutions/Enforcing-Naming-Convention (github.com)
     - This code is similar to the above but uses local data parallelism (torch.nn.parallel.DistributedDataParallel) to speed up the training process - Enforcing-Naming-Convention/local_data_parallelism.py at main · People-Places-Solutions/Enforcing-Naming-Convention (github.com)
     - This code is similar to the above but uses data parallelism for colab notebook or any other cloud based computing (torch.nn.DataParallel) to speed up the training process - Enforcing-Naming-Convention/local_data_parallelism.py at main · People-Places-Solutions/Enforcing-Naming-Convention (github.com)
     - This code uses DeepSpeed to speed up - Enforcing-Naming-Convention/single_gpu.py at main · People-Places-Solutions/Enforcing-Naming-Convention (github.com)
   - LSTM model building
     - Not a good idea, I tried on google colab / replit and the entire notebook kept crashing. https://colab.research.google.com/drive/10yCkBVf48_NYZJVTb-NED_9lscywTpj2?usp=sharing
       I think it's better to download it and open on VSCode or alike
   - Fuzzy Matching
     - Perhaps the easiest way so far
     - Uses TheFuzz - Python lib
       Enforcing-Naming-Convention/fuzz_trial.py at main · People-Places-Solutions/Enforcing-Naming-Convention (github.com)
   - Using GPT 4o model API to generate incorrect variations of names for training the models
     - This is there inside the LSTM code
     - An API Key is required. I think free public keys can be acquired for limited tokens

2) **Inertia Calculator** (WPF)

Repo link: **People-Places-Solutions/RFT-Inertia-Application: WPF Application calculating Inertia for Circular, Rectangular sections (more to be added) (github.com)**

Dependencies for this entire project:

- PdfiumViewer
- MahApps.Metro
- Microsoft.Toolkit.Uwp.Notifications
- PropertyChanged.Fody
- System.Drawing.Common
- Jacobs.Orchestra.UI

- Dynamic Canvas
    - Placed in a separate class
      RFT-Inertia-Application/WpfApp1/ViewModel/Dynamic Canvas at main · People-Places-Solutions/RFT-Inertia-Application (github.com)
- Report Preview Window
    - Code behind has the logic
    - DocumentViewer was used to display the document
      RFT-Inertia-Application/WpfApp1/View/ReportWindow.xaml at main · People-Places-Solutions/RFT-Inertia-Application (github.com)
    - Two classes introduced for the workflow Visuals - > XPS -> PDF
      RFT-Inertia-Application/WpfApp1/ViewModel/DocumentConverter.cs at main · People-Places-Solutions/RFT-Inertia-Application (github.com)
      RFT-Inertia-Application/WpfApp1/ViewModel/MyDocumentPaginator.cs at main · People-Places-Solutions/RFT-Inertia-Application (github.com)

      DocumentPaginator: It creates a multipage document from a single page document. It assigns page numbers equal to number of the visuals (hold on to this thought – this page number is not the final page number). There is an option of choosing Landscape and Portrait. This is now passed to DocumentConverter.

      DocumentConverter: Creates a XPS document. This creates the final look and feel of the doc. Headings, subheadings, margin size, font styles, borders are set here. Here the page numbers from DocumentPaginator are used as a count of number of visuals to be placed. The size (height, width) of the visual is calculated based on the number of the visuals to be placed on each document page.

3) **ETABS Visualiser** (Streamlit)

Repo link: **People-Places-Solutions/ETABS-Visualisation-Streamlit (github.com)**
Hosted website: **etabs-workbook-visualisation.streamlit.app**

- Streamlit Dashboard
    - Easy to code, rapid prototyping
    - One issue is that the entire page refreshes with every button click (radio buttons, checkbox, any other control). Look up about session states to preserve selected options or uploaded files

4) **ETABS Visualiser** (Shiny)

Repo link: **People-Places-Solutions/ETABS-Visualisation-Shiny (github.com)**
Hosted website: **riddhi-goswami.shinyapps.io/etabsvisualisation/**

- Shiny Dashboard
    - Harder compared to Streamlit
    - Main thing to look up is render, reactive decorators
    - Good documentation when looking up its API information (Shiny for Python (posit.co))
    - Deploying: Shiny for Python - Cloud hosting (posit.co)
- About the code
    - There are functions defined for each plot type, merging the graphs, joining the datasets. I've uploaded the python notebook – showing the thought process behind it.

Recommended Reading *(it's literally my course work XD)*

- Machine Learning (Tom M. Mitchell)
- Artificial Intelligence: A Modern Approach (Stuart, Peter)
- Introduction To Algorithms (Thomas H. Cormen, ...)
- Introduction to Data Mining (Pang-Ning Tan)
- Statistics for Business & Economics (James J. Cochran, …)
