Project 1: AI Powered Task Management System

**Project Summary**: Al-Powered Task Management System Goal: Automatically classify and prioritize tasks using machine learning (ML) and natural language processing (NLP).

**Steps**:

Collect data → Clean it → Preprocess →Train models → Evaluate →

Feature extraction.

**Dashboard**: Task follow to make the project.

* Data Collection
* EDA + data cleaning
* NLP
* Feature Extraction
* Task classification

**Person 1**- Nikeeta

* **Task** : Dataset Collection + Raw Data Storage
* **Technology used**: Trello, Jira API, Python

**WHAT TO DO?**

Dataset Collection & Storage. This means that Nikeeta will have to collect task-related data - this data can be from the API of Trello or Jira, or it can be obtained from an open-source platform like Kaggle. If original data is not available, then synthetic (manufactured) task data can also be generated. After that, the data obtained will have to be stored in CSV or JSON format in a proper way so that other team members can use it easily. In this process, the user will have to fetch data from APIs using Python (requests library). This work can be done on VS Code or Google Colab, depending on convenience. After collecting the data, that file will be given to Tanisha (EDA team member) for analysis

**Person 2**- Tanisha

* **Task** : Data Cleaning
* **Technology**: Python, Pandas, matplotlib, seaborn

**WHAT TO DO**?

Tanisha has to work on Google Colab or Jupyter Notebook where she will open the CSV data provided by Nikeeta using Python language. If there are any missing values, wrong format, or extra columns in that data, she will clean them up - for example, if the priority of a task is missing, fill it with fill() or remove it with drop(). After cleaning, she will analyze the data, in which she will see how many tasks are assigned to which person, how many tasks are of which priority (High, Medium, Low), and the deadlines are till what date. Then she will show the analysis in the form of a graph using matplotlib or seaborn, so that the team can understand the data clearly

**Person 3 – Lavanya**

* **Task** : NLP Preprocessing
* **Technology**: Python + NLP libraries (NLTK, spacy, re )

**WHAT TO DO?**

Lavanya will perform text cleaning on the task descriptions cleaned by Tanisha - like converting to lowercase, removing punctuation, removing stopwords, and lemmatization/stemming. After that, Lavanya will create features from the same cleaned text data using TF-IDF or Word2Vec, which the machine learning model will use. They have to perform this task in google collab using python and NLP libraries.

Person 4 – Avi

* **Task** : Feauture Extraction
* **Technology** : Python, scikit-learn

**WHAT TO DO?**

Avi's job is to do feature extraction, that is, convert the text data cleaned by Tanisha and Lavanya into a numerical format which can be used by the machine learning model. For this, Avi will use Python and the platform will be Google Colab or Jupyter Notebook. Avi will use TF-IDF Vectorizer of scikit-learn in which each task description will be converted into a numerical vector. If more advanced features are required, Word2Vec of gensim can also be used. These features created by Avi are in the form of an array (X) which the next team member (Lavisha) will use as input to the machine learning model. In this process, Avi has to take text data, vectorize it using TF-IDF, and prepare clean numerical data as output for the model.

Person 5 – Lavisha

* **Task** : Task classification
* **Technology** : python, scikit-learn

**WHAT TO DO?**

Lavisha's job is to take the numerical features (TF-IDF vectors) created by Avi and create Machine Learning models, train them, and test them - such as Naive Bayes or Support Vector Machine (SVM) models. After that, she will evaluate the performance of the models (by looking at accuracy, precision, recall, confusion matrix.

**Model Saving & Integration**

Save the trained model of Lavisha in .pkl or .joblib format - so that it can be used on the web or in a mobile app. Then we will save the model with joblib.dump(model, "model.pkl")