1. What are the names and NetIDs of all your team members? Who is the captain? The captain will have more administrative duties than team members.

   Pinky Chauhan (pinkyc2)
   I will be working on this project individually. All administrative work will be done by me.

2. What is your free topic? Please give a detailed description. What is the task? Why is it important or interesting? What is your planned approach? What tools, systems or datasets are involved? What is the expected outcome? How are you going to evaluate your work?

   Topic: Fake news classification using machine learning

   Details/Task:
   I plan to build a system that is able to tell apart fake news from real news given some title and/or news content as the input.
   This is essentially a classification problem where I will train several models using machine learning on the following dataset from Kaggle:
   https://www.kaggle.com/hiteshkumargupta/fake-news-classification.
   These trained models will then predict the category of news item from a test dataset of fake/real news articles.
   Models will be evaluated based on performance metrics to choose the final model that will be used for predictions in the classifier script/API for final submission.

   Why fake news classifier?
   Fake news is becoming increasingly prevalent nowadays especially with the wide-spread usage of social media platforms which can be easily misused to propagate factually incorrect information to the users.
   With an average person spending many hours in each day coming across multiple posts, tweets, news articles, etc. while on social media, it becomes important to be able to segregate actual facts from cooked up fake stories/news.
   Such a tool can then be integrated with social media platforms to flag such articles or filter those out.
   It is an interesting problem to solve since fake news articles can come very close to the tone or style of the real news to make it sound authentic and hence not very easy to identify.

   Planned approach:
   I plan to divide the project into the following steps:

   - Data Analysis to understand the observations listed in the dataset and their contribution towards the classification.
   - Preprocessing to setup training and test datasets, handle missing values, performing tokenization, remove stop words, stemming, encode categorical variables.
   - Feature Selection to keep only the most relevant variables
   - Vectorization to map words to a corresponding vector of real numbers to find word similarities, etc.

- Model design to train, tune hyperparameters, validation, test using several classification algorithms (XGBoost, Naive-Bayes, Decision tree, Linear classification, SVM, etc.)
- Performance evaluation: compute and analyze the metrics precision, recall, F1 score, etc.
- Create API/script that will take news text as input and generate it's classification as real or fake as the result.

Tools/systems/datasets:
I plan to leverage nltk for preprocessing tasks, numpy and pandas, sklearn for machine learning, matplotlib, etc. for this project.
Dataset: https://www.kaggle.com/hiteshkumargupta/fake-news-classification

Expected outcome:
An API/python script/Jupiter notebook that can accept news title and/or text as input and output the category for the news item as fake news or real news.

Evaluation:
I will be training several models using different algorithms and evaluate their performances on the test dataset using precision, recall, F1 score, etc.

3. Which programming language do you plan to use?
   Python/Jupiter notebook

4. Please justify that the workload of your topic is at least 20*N hours, N being the total number of students in your team. You may list the main tasks to be completed, and the estimated time cost for each task.

Task	Estimated Time
Understand classification algorithms in depth and familiarize with nltk (I am new to machine learning world and will need to research/obtain a deeper understanding)	8 hours
Environment setup	1 hour
Data analysis and preprocessing	3 hours
Feature selection and vectorization	2 hours
Model design, training and hyperparameters tuning	10 hours
Testing and evaluation	4 hours
Integration with final output script/API	1 hour
Prepare presentation	2 hours
Total	**31 hours**