**1 - User Interface**

* Simple with big text field
* Tasks:
  + Design the interface (html?)
  + Design FindAnswer function so we can decide what is the best input or how the parsing should be made
  + Decide how are we going to parse/tag
  + Implement parse/tagging method
  + Implement FindAnswer and DBQuery functions (?)
  + Print the output (simple answer, table, calculation, etc)

**2 - DB of Questions and Queries**

* 1 table, 3 fields: QuestionID, Question and SQL Statement
* Tasks:
  + Create de DB
  + Input a variate set of questions/statements
  + Test it with different questions

**3 - Data**

* 2 tables: grades and attendance (should we do attendance?)
* Grades table: StudentID, Name, Class, AssignmentName, AssigmentGrade, AssignmentDueDate
* Attendance table: StudentID, WeekNumber, Date, Attendance
* Tasks:
  + Create DB and input data

**4 - Workflow overview:**

1. User types question
2. Parse and tag the question
3. Call function FindAnswer
4. FindAnswer funcion lookup for match in the QQDB, and retrieve the SQL query
5. DBQuery function runs the SQL statement and retrieve the result(s)
6. Print result(s) to user

Division of labor:

Fernando: NLP - synonym

After the template is selected I should know the parameters I should look for

find different ways the user may refer to parameters and output it in a way the translation will understand (Ass1, Assignment 1, Assignment one, etc)

Sneha: user interface + poster

Wenzhe: NLP - parsing and selecting

Yu: all background coding

1.Files:

* 1. questionbank.txt: stores a list of the types of questions that can be asked, one per line
  2. data.csv: stores the students’ information and grades, comma separated table
  3. database.py: able to load data.txt on initialization, contains functions to select and calculate given query
  4. parse.py: able to load questionbank.txt on initialization, contains functions to interpret english input questions and match to question bank
  5. translate.py: used to translate from output of parse.py into query for database.py
  6. interface.py: contains the interface to interact with user
  7. main.py: this is the main code that connects everything else

2. User interface:

A simple text-based, line-by-line input/output in python console

3. Work flow:

1. user runs main.py, which initializes database.py and parse.py. main.py outputs instructions to use program, shows ready status
2. user types in question
3. question passed to parse.py, which find matching question model, then outputs “question” object
4. “question” object passed into translate.py, which translates into “query” object
5. “query” object given to database.py, which returns “result” object
6. based on type of question and result, main.py outputs result and waits for next question
7. return to step b.

4. Dataset: 50 students

1. student number (unique) (or netID?)
2. first name
3. last name
4. gender?
5. assignment grades x 10
6. exam grades x 2

5. Questions:

1. What is the class average?
2. What is the class average for assignment \_?
3. What is the class average for exam \_?
4. What is <student name>’s average?
5. What is <student name>’s score for <assignment or exam>?
6. Of those who <criteria>, what is their <assignment or exam score>?
7. Who had <highest, lowest> score on <assignment, exam, or average>?
8. How many students had <criteria>?
9. Is there a <student name> in this class?
10. What is <student first/last name>’s last or first name?