

Name: Claire Chen
Andrew ID: ccz
Section C

15-112 Term Project Proposal

- **Project Description**

MyCrispr is an app that generates structures of CRISPR cas-9 enzymes that can specifically target the user-generated DNA sequence and edit the sequence. With this app, users can select their sequence of interest, visualize the structure of the program-generated CRISPR cas-9 enzyme that is potentially capable of editing the gene, and compare the structure of proteins translated from the sequence before and after the edit.

Post-MVP, additional goals such as homology modeling and algorithm visualizer (i.e. showing the process of generating the enzymatic structure) will be pursued as part of this project.

- **Competitive Analysis**

There is a wide array of existing protein folding and visualization software that can interpret source data to produce three-dimensional protein models. For example, PyMol is a molecular graphics software that visualizes proteins, nucleic acids, and small molecules as well as allows users to edit their molecules. Similar to PyMol, my project will involve visualizing proteins and nucleic acids and allowing users to rotate the structures and zoom in and out while visualizing them. My program will read and parse PDB (protein data bank) protein source files and convert the data into strings, which is also how PyMol obtains its protein modeling data.

Another example is Foldit, an interactive puzzle game that involves protein folding. Players can use a variety of tools to optimize their design of protein structure, contributing to results that outperform computer-generated structures, giving rise to potential for accelerating scientific discovery. Although users of my completed project won't be able to interact with the protein product directly through folding secondary structures and changing the protein conformations, they can visualize firsthand how their original protein structure produced by their input DNA sequence changes as a result of changing the DNA template sequence and the use of the generated CRISPR cas-9 enzyme.

One distinct difference between PyMol, Foldit, and my project is taking gene editing instructions from the user directly, as opposed to simply reading off protein source files. My project will involve taking DNA input generated directly by the user, comparing the expected protein product against existing protein structures to determine the potential structure of specific CRISPR cas-9. The mechanism for changes in protein structures is user-generated DNA edits rather than direct interaction with the protein structure.

- **Structural Plan**

I plan to have the following files in my project directory.

main.py	The location of all majors functions and contains the blueprint of the project. All functions and variables in other files are referenced in this file.
parser.py	A file for parsing PDB (protein data bank) files
seqAnalysis.py	For translating nucleic acid sequences into amino acid sequences
webscraping.py	Obtains more information from websites to be displayed to the users after they input genetic sequences (e.g. species to which the most similar proteins belong)
projections.py	Stores the OOP functions and coordinate system conversion functions that are used in main.py

Potentially more files will be added based on the length of the code in each file.

- **Algorithmic Plan**

There are two major ways in which this project is algorithmically complex.

First, protein modeling.

My first step of protein modeling involves isometric rendering of proteins, given the x, y, z coordinates of individual atoms that constitute a protein by parsing a PDB file. This utilizes knowledge in linear algebra, including linear transformations, projection matrices, and norms. My next step is to model protein structures using alpha helices and beta pleated sheets by determining the start and end of each secondary protein structure based on the amino acid and atomic sequence of a protein in a PDB file. The challenge with this step lies in using actual 3D modeling techniques rather than isometric projections. Since secondary protein structures resemble coils, ribbons, and complex structures that need to be rendered using more computationally intensive methods and potentially modules such as Pandas3D.

Second, generation of CRISPR cas-9 enzyme.

Post-MVP, my goal is to incorporate machine learning into my project by training models on existing gene editing data (vector systems and structures of CRISPR cas-9 enzymes with their corresponding gene edits). I hope to produce CRISPR cas-9 enzyme structures that can be experimentally tested in the lab to help scientists solve the CRISPR cas-9 enzymes for gene editing in a more efficient manner. This component of the project also aims to help those who are interested in the technology understand the underlying mechanisms of CRISPR-enabled genetic engineering.

- **Timeline Plan**

By the tech demo, I was able to demonstrate my understanding of the following: simple isometric rendering of atoms in a protein, PDB file parsing, scaling RGB values of atoms (shading) to show depth.

By TP1, I can achieve 3 dimensional rotation of protein structures along 3 axes and in two other directions (left and right) in the viewer's perspective. I am able to use a custom comparator function to sort through a list of atoms as instances of the object Protein (in OOP) based on the distance between individual atoms and the viewer by taking the norm of the vectors formed between the coordinates.

By TP2, I aim to complete all major features pertaining to protein modeling, including non-isometric 3D modeling of proteins (rendering alpha helices and beta pleated sheets). At this stage, I hope to get started on machine learning, experimenting with modules, determining the optimal type of ML models, constructing models, and training the model using data.

- **Version Control Plan**

I use Git to update changes to my project files. After creating a private GitHub repository called "112-Term-Project," I cloned it to my local computer. My plan is to commit and push changes to Git on a daily basis and view the editing history as needed. You can find screenshots of my terminal and Github repository below to see how I use Git for version control.

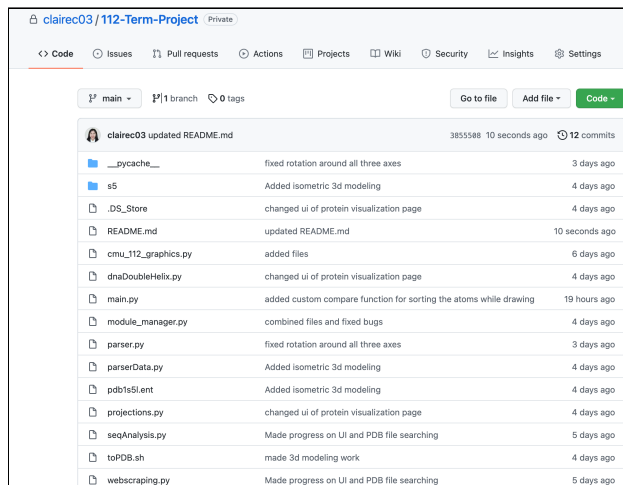
```
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
clairechen@Claire-MacBook-Air:112-Term-Project % git status
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   main.py

no changes added to commit (use "git add" and/or "git commit -a")
clairechen@Claire-MacBook-Air:112-Term-Project % git add
clairechen@Claire-MacBook-Air:112-Term-Project % git commit -m "added custom compare function for sorting the
atoms while drawing"
[main e32c48c] added custom compare function for sorting the atoms while drawing
1 file changed, 182 insertions(+), 38 deletions(-)
clairechen@Claire-MacBook-Air:112-Term-Project % git push
Enumerating objects: 5, done.
Counting objects: 388 (0/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 1.68 MiB | 1.68 MiB/s, done.
Total 3 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/clairec03/112-Term-Project.git
  61d79e1..e32c48c main -> main
clairechen@Claire-MacBook-Air:112-Term-Project % git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
clairechen@Claire-MacBook-Air:112-Term-Project %
```



File	Commit Message	Time Ago
clairec03 updated README.md	3855588	10 seconds ago
__pycache__	fixed rotation around all three axes	3 days ago
s5	Added isometric 3d modeling	4 days ago
_DS_Store	changed ui of protein visualization page	4 days ago
README.md	updated README.md	10 seconds ago
cmu_112_graphics.py	added files	6 days ago
dnaDoubleHelix.py	changed ui of protein visualization page	4 days ago
main.py	added custom compare function for sorting the atoms while drawing	19 hours ago
module_manager.py	combined files and fixed bugs	4 days ago
parser.py	fixed rotation around all three axes	3 days ago
parserData.py	Added isometric 3d modeling	4 days ago
pdbtSlient	Added isometric 3d modeling	4 days ago
projections.py	changed ui of protein visualization page	4 days ago
seqAnalysis.py	Made progress on UI and PDB file searching	5 days ago
toPDB.sh	made 3d modeling work	4 days ago
webscraping.py	Made progress on UI and PDB file searching	5 days ago

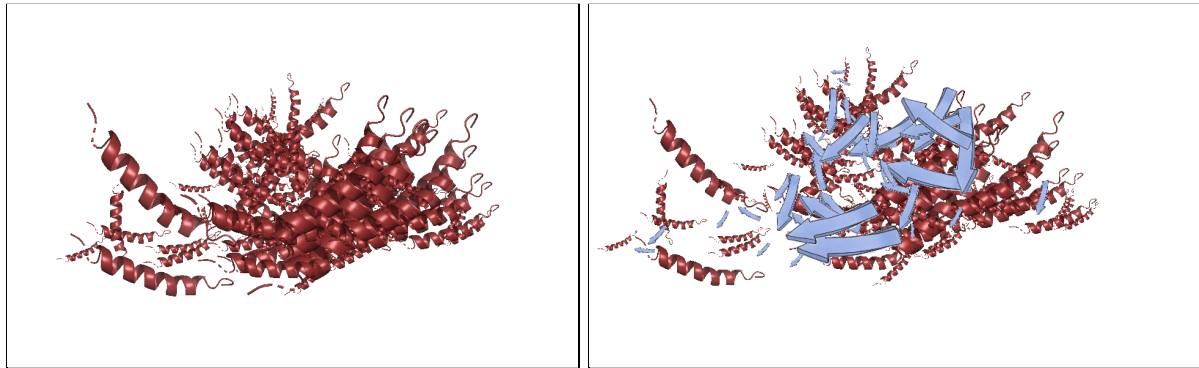
- **Module List**

A list of the modules that are currently in use (approved as part of passing the tech demo) include NumPy and BioPython. I intend to use additional modules such as TensorFlow, Keras, and Pandas for determining the structure of specific CRISPR cas-9 enzymes using machine learning as well as BeautifulSoup for web scraping, since it can help retrieve more information about molecules/DNA sequences that are not available in the Protein Data Bank.

- **Updates Since TP1 (Nov 22)**

1. **parser.py**: After learning to read the secondary structures of proteins based on their PDB files, I created a dictionary for secondary structures (both alpha helices and beta sheets)

2. **parser.py:** I created a dictionary that enumerates individual amino acid sequences with keys being their indices and values being the coordinates of atoms that constitute the amino acids
3. **secondaryStruct.py:** I am able to use coordinates of the starting and ending points of secondary structures and use scaled png images for drawing alpha helices and beta sheets.



- **Updates Since TP2**

1. **getUserInput:** I fully integrated PDB file retrieval using BioPython with the app that runs on main.py. Users can now click on the button “Choose your protein” to enter a 4-digit PDB code. My Python file would then request the PDB file from the Protein Data Bank and download it to the local directory for protein visualization and editing when prompted.
2. **DNA editor:** I created a new UI that supports edits of the DNA of protein that the user requests by entering the 4-digit PDB code. Users can manually insert DNA sequences or delete nucleotide bases by using the scissors and pencil tools on the DNA visualization page. Given the intended audience (presumably students or researchers with prior scientific knowledge), I decided to display experiment-related information, including the melting temperature and feedback on the genetic edits (type of mutation) to inform the users.
3. **DNA scroller:** Users can easily edit the DNA of the protein by using the left or right arrow keys to scroll through the DNA strand. There is a yellow triangle-shaped pointer that displays the current location of the base along the DNA sequence that encodes the desired protein as well as highlighter shape that moves along the double helix as the user moves to the left or right of the DNA sequence.
4. **Integration of atomic view of proteins and view of secondary structures:** On the protein visualizer page, users can now toggle between secondary structures and atomic model of the protein.
5. **Rotation of secondary structures:** Instead of overlaying scaled images of beta sheets that illustrate the approximate starting and ending locations of each sheet, beta-pleated sheets are now drawn with smoothed lines using `canvas.create_line()`. This function passes in a list of tuples so the accurate beta-sheets can be projected onto the screen. Secondary structures are color coded based on their indices in the list of beta sheet coordinates stored as `app.sheetStruct`.

6. **Display of additional information about the protein being visualized:** The app now retrieves more information by parsing the requested PDB files and displays more protein information on the 3D visualization page.
7. **Updated structural plan:** Locating functions in individual files presented a challenge for me. I decided to pivot to a new structural plan of integrating all files into one main.py for easier navigation. I simply use the Command + G search tool in the main.py file whenever I need to look for a specific function.