# Εργασία στο μάθημα Τεχνολογία Λογισμικού - Εφαρμογή Streamlit για Ανάλυση Δεδομένων Μοριακής Βιολογίας

Μπράχμα τζωρτζ Σαβαδράμ - inf2022176

## Περιεχόμενα

**Περίληψη**

Η παρούσα εργασία περιγράφει τον σχεδιασμό και την υλοποίηση μιας εφαρμογής Streamlit που υποστηρίζει εξερεύνηση δεδομένων και εργασίες μηχανικής μάθησης σε σύνολα δεδομένων μοριακής βιολογίας. Η εφαρμογή επιτρέπει στους χρήστες να ανεβάζουν και να οπτικοποιούν δεδομένα, να πραγματοποιούν συσταδοποίηση (clustering) με χρήση της μεθόδου KMeans, να εκπαιδεύουν μοντέλα ταξινόμησης (Random Forest), και να κάνουν προβλέψεις με ήδη εκπαιδευμένα μοντέλα. Η εφαρμογή είναι σχεδιασμένη να είναι προσβάσιμη, επεκτάσιμη και φορητή μέσω Docker.

# 1 Εισαγωγή

Η ανάλυση δεδομένων και η μηχανική μάθηση αποτελούν πλέον αναπόσπαστο κομμάτι της βιοπληροφορικής και της μοριακής βιολογίας, όπου μεγάλες ποσότητες δεδομένων προκύπτουν από μελέτες όπως η πρόβεψη δομών προτεΐνων, η αλληλούχιση DNA, και η ταξινόμηση δειγμάτων. Αυτή η αύξηση των διαθέσιμων δεδομένων απαιτεί εργαλεία που να επιτρέπουν γρήγορη οπτικοποίηση, κατανόηση και πρόβλεψη.

Η εφαρμογή που παρουσιάζεται σε αυτή την εργασία δημιουργήθηκε με σκοπό να παρέχει ένα ολοκληρωμένο, διαδραστικό περιβάλλον για την ανάλυση δεδομένων μοριακής βιολογίας. Ο χρήστης έχει τη δυνατότητα να:

- ανεβάσει τοπικά δεδομένα (CSV) και να τα εξερευνήσει με στατιστικά και γραφήματα,
- εφαρμόσει αλγορίθμους clustering (KMeans) για την εύρεση μοτίβων,
- εκπαιδεύσει μοντέλο ταξινόμησης (Random Forest),
- φορτώσει εκπαιδευμένο μοντέλο και να πραγματοποιήσει προβλέψεις,
- δει συγκριτικά αποτελέσματα και ποσοστά ακρίβειας.

Η υλοποίηση πραγματοποιήθηκε σε Python και βασίστηκε στους κώδικες του μαθήματος στο opencourses. Τα παραδείγματα που παρουσιάζονται στα σχήματα 3 - 5 προέκυψαν από το excel dataset ”iris” και από το, ήδη εκπαιδευμένο μοντέλο: ”saved-model.pkl” τα οποία δωθήκανε από το μάθημα. Η εφαρμογή είναι πλήρως Dockerized, ώστε να μπορεί να εκτελεστεί σε οποιοδήποτε περιβάλλον χωρίς επιπλέον ρυθμίσεις ή εγκαταστάσεις. Το έργο συνοδεύεται από πλήρη τεκμηρίωση, UML διαγράμματα, οπτικοποιήσεις και link σε αποθετήριο GitHub.

github.com/Citypop15/inf2022176$_{}streamlit$

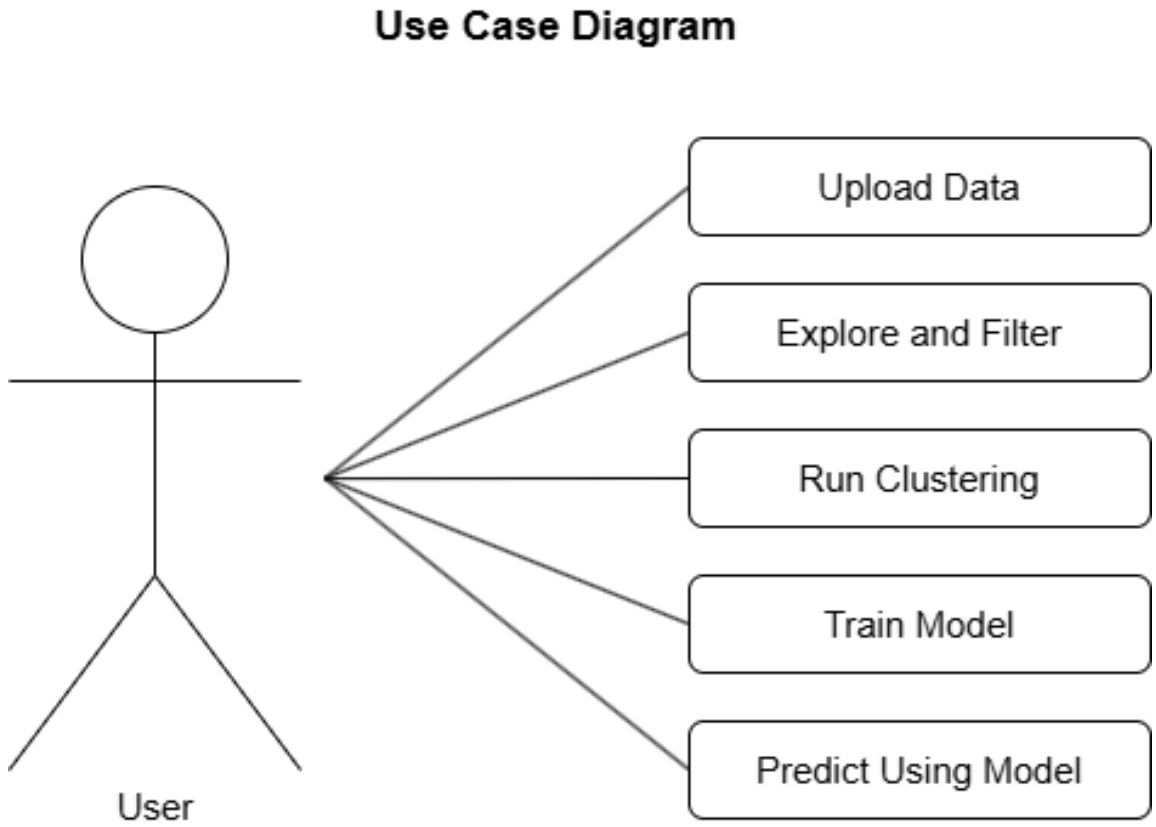
# 2 Σχεδιασμός Υλοποίησης

Η εφαρμογή αποτελείται από ξεχωριστές λειτουργικές ενότητες, καθεμία από τις οποίες προσπελαύνεται μέσω του sidebar στην Streamlit:

- Upload and Explore: Επιτρέπει την μεταφόρτωση αρχείων CSV, προβολή δεδομένων, περιγραφική στατιστική, φιλτράρισμα με λέξεις-κλειδιά, και γραφική αναπαράσταση scatter plot.
- Clustering: Υλοποιεί συσταδοποίηση με KMeans. Ο χρήστης μπορεί να επιλέξει τον αριθμό των clusters και εμφανίζονται τα αποτελέσματα με σχετική οπτικοποίηση.
- Train Model: Πραγματοποιείται επίβλεψη ταξινόμησης με Random Forest. Τα δεδομένα χωρίζονται σε train/test και εμφανίζεται η ακρίβεια. Η εκπαίδευση του μοντέλου πραγματοποιείται στο παρασκήνιο. Εδώ εμφανίζεται μόνο η ακρίβεια (accuracy) του Random Forest.
- Predict: Επιτρέπει την εισαγωγή προ-εκπαιδευμένου μοντέλου (.pkl) και την εφαρμογή του σε νέα δεδομένα. Εάν υπάρχουν ετικέτες (labels), συγκρίνεται η πρόβλεψη με την πραγματικότητα.
- Creator Info: Πληροφορίες δημιουργού.

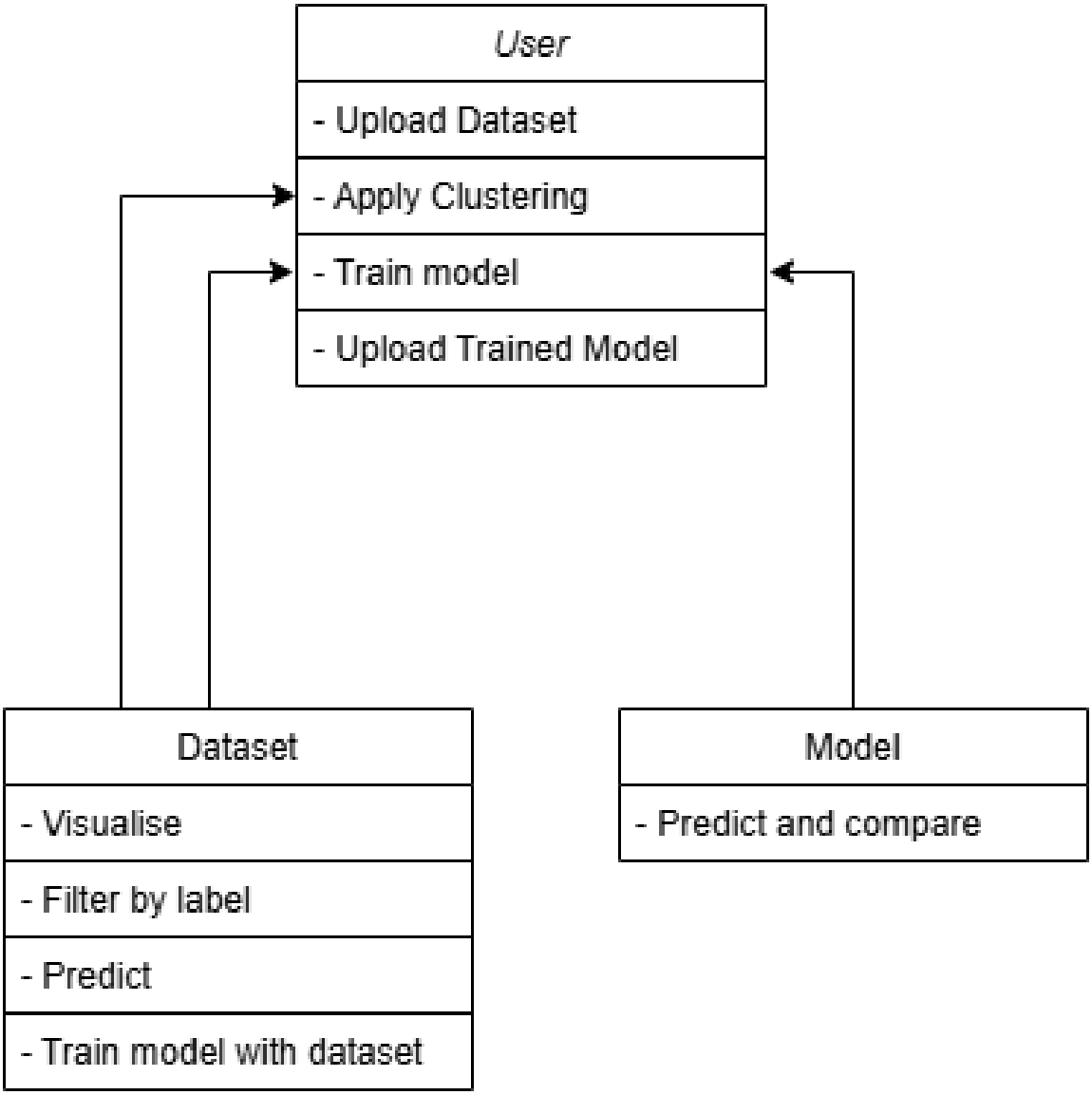
Τα παρακάτω διαγράμματα UML δημιουργήθηκαν με χρήση του δωρεάν εργαλείου `draw.io`.

## 2.1 Σχήμα 1: Use Case Diagram




Σχήμα 1: Use Case διάγραμμα για την αλληλεπίδραση χρήστη-εφαρμογής

### 2.2 Σχήμα 2: Class Diagram




Σχήμα 2: Class διάγραμμα που απεικονίζει τη ροή δεδομένων μέσα από τα μέρη της εφαρμογής

Για τα τμήματα: ”Upload and Explore”, ”Clustering και ”Train model”, ο χρήστης χρειάζεται να ανεβάζει μόνο κάποιο dataset, (iris σε αυτή την περίπτωση), ενώ στο τμήμα predict πρέπει να ανεβάσει και ένα προ-εκπαιδευμένο μοντέλο (.pkl).

## 3 Ανάλυση της Υλοποίησης

Ο κώδικας της εφαρμογής αναπτύχθηκε ενσωματώνοντας τα περιεχόμενα των αρχείων myapp1.py έως myapp9.py του μαθήματος. Αναλυτικά:

- Το myapp1.py χρησιμοποιήθηκε για την υλοποίηση μεταφόρτωσης αρχείων και προβολής δεδομένων.
- Τα myapp2.py και myapp3.py εισήγαγαν τη λειτουργία clustering με KMeans και επιλογή αριθμού clusters.

- Το myapp4.py χρησιμοποιήθηκε για επιλογή αξόνων X και Y σε scatter plots.
- Το myapp5.py παρείχε ιδέες για την αισθητική διάταξη του περιβάλλοντος.
- Το myapp6.py έδειξε χρήση sidebar για παραμέτρους χρήστη.
- Το myapp7.py έδωσε τη δυνατότητα φιλτραρίσματος με βάση τιμές/λέξεις.
- Το myapp8.py χρησιμοποίησε Random Forest για ταξινόμηση.
- Το myapp9.py περιείχε τη διαδικασία εισαγωγής εκπαιδευμένου μοντέλου και πρόβλεψης.

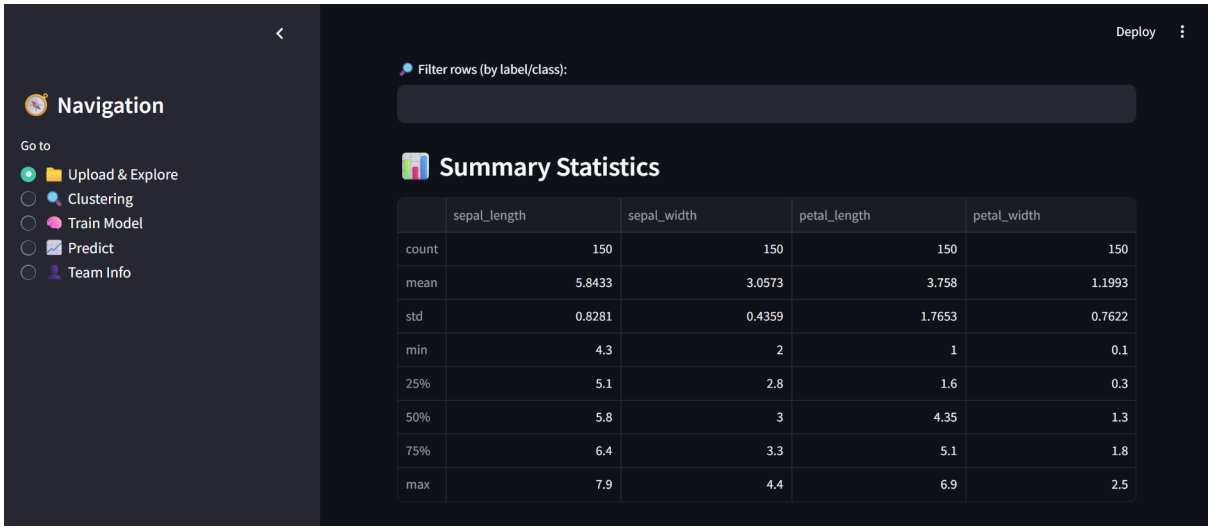
Όλες οι παραπάνω λειτουργίες συνδυάστηκαν και προσαρμόστηκαν σε έναν ενιαίο κώδικα app.py αντί για πολλαπλά .py αρχεία, ο οποίος οργανώνεται με if συνθήκες βάσει επιλογής από το sidebar. Η Streamlit επιτρέπει την υλοποίηση διαδραστικής εφαρμογής μέσω Python χωρίς ανάγκη HTML ή JavaScript. Η διεπαφή της εφαρμογής οργανώθηκε ως εξής:

- Ο χρήστης μετακινείται ανάμεσα στις καρτέλες μέσω st.sidebar.radio(...).
- Με st.file-uploader(...), μεταφορτώνει δεδομένα ή μοντέλα.
- Η προβολή δεδομένων γίνεται με st.dataframe(...) και st.write(...).
- Ο χρήστης επιλέγει παραμέτρους με st.slider(...), st.selectbox(...), st.text-input(...).
- Η γραφική αναπαράσταση γίνεται με st.plotly-chart(...).

Επίσης, η γραμματοσειρά και τα χρώματα της εφαρμογής καθορίζονται από το αρχείο config.toml που βρίσκεται στον φάκελο ”.streamlit”.

# 4 Οπτικοποιήσεις Αποτελεσμάτων

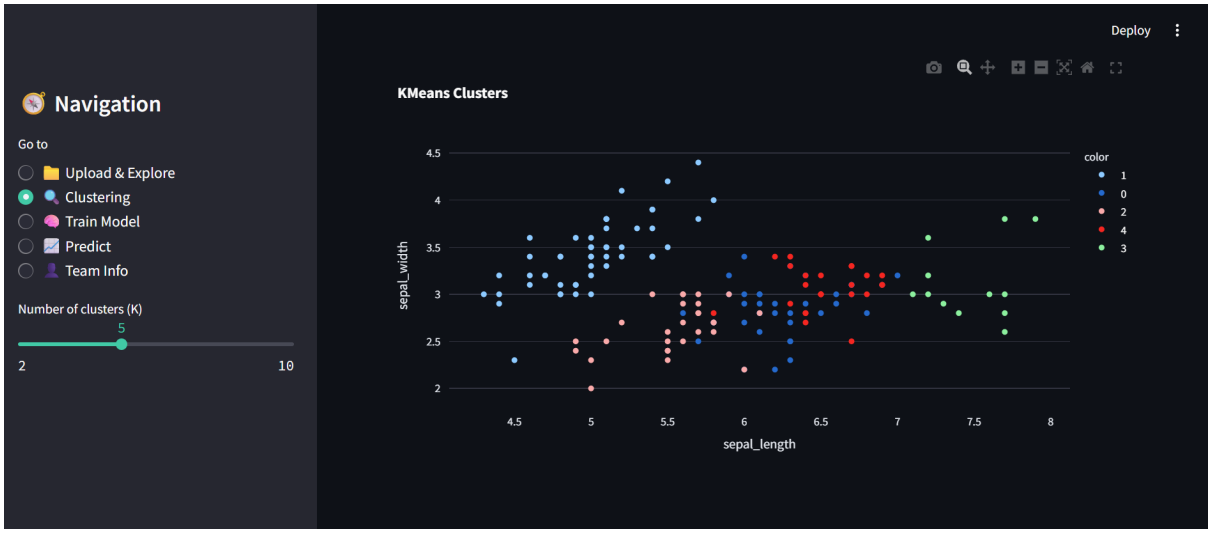
## 4.1 Σχήμα 3: Upload Explore




| | sepal_length | sepal_width | petal_length | petal_width |
|---|---|---|---|---|
| count | 150 | 150 | 150 | 150 |
| mean | 5.8433 | 3.0573 | 3.758 | 1.1993 |
| std | 0.8281 | 0.4359 | 1.7653 | 0.7622 |
| min | 4.3 | 2 | 1 | 0.1 |
| 25% | 5.1 | 2.8 | 1.6 | 0.3 |
| 50% | 5.8 | 3 | 4.35 | 1.3 |
| 75% | 6.4 | 3.3 | 5.1 | 1.8 |
| max | 7.9 | 4.4 | 6.9 | 2.5 |

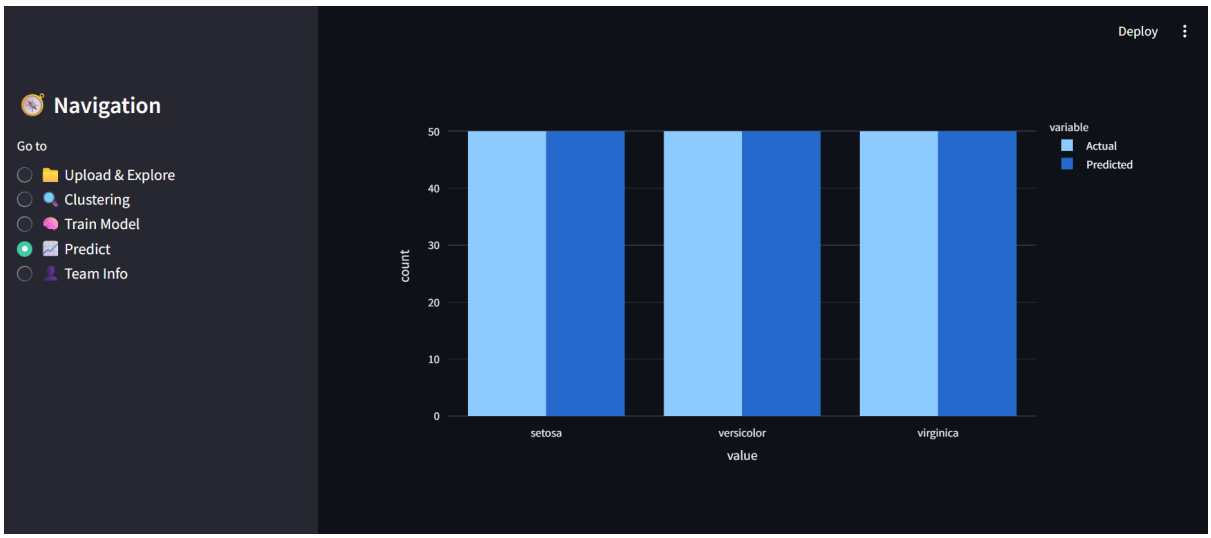
Σχήμα 3: Upload & Explore: Σε αυτό το στάδιο, ο χρήστης μπορεί να ανεβάσει αρχείο CSV με τα βιολογικά δεδομένα του. Η εφαρμογή προβάλλει αμέσως έναν πίνακα με τις πρώτες γραμμές των δεδομένων και εμφανίζει βασικά στατιστικά για κάθε στήλη: μέσο όρο, ελάχιστη/μέγιστη τιμή, τυπική απόκλιση κ.λπ. Ο χρήστης έχει τη δυνατότητα να φιλτράρει τις γραμμές πληκτρολογώντας έναν όρο αναζήτησης (π.χ. ”versicolor”) και να επιλέξει δύο μεταβλητές (στήλες) για δημιουργία διαδραστικού scatter plot.

## 4.2 Σχήμα 4: Clustering




Σχήμα 4: Clustering: Ο χρήστης επιλέγει ένα αρχείο CSV με μη επισημασμένα δεδομένα και χρησιμοποιεί το αλγόριθμο KMeans για να δημιουργήσει συστάδες (clusters). Ο αριθμός των clusters καθορίζεται δυναμικά από τον χρήστη μέσω slider. Η εφαρμογή εφαρμόζει το clustering και προσθέτει μια νέα στήλη στο dataframe με την ομάδα κάθε δείγματος. Έπειτα, ο χρήστης επιλέγει ποιες στήλες θα εμφανιστούν στο scatter plot. Το γράφημα δείχνει πώς τα δεδομένα ομαδοποιούνται σε ξεχωριστές περιοχές, αποκαλύπτοντας μοτίβα ή κατηγορίες.

## 4.3 Σχήμα 5: Prediction




Σχήμα 5: Prediction: Η καρτέλα πρόβλεψης επιτρέπει την εισαγωγή εκπαιδευμένου μοντέλου (αρχείο .pkl) και ενός test dataset. Η εφαρμογή εφαρμόζει το μοντέλο και δημιουργεί προβλέψεις για την τελική στήλη (π.χ. κατηγορία δείγματος). Αν το αρχείο περιλαμβάνει την πραγματική ετικέτα, γίνεται αυτόματη σύγκριση και υπολογίζεται ακρίβεια (accuracy). Τα αποτελέσματα εμφανίζονται τόσο σε πίνακα όσο και σε συγκριτικό γράφημα bar chart, δίνοντας εικόνα για την απόδοση του μοντέλου.

# 5 Dockerization της Εφαρμογής

Η χρήση του Docker επιτρέπει τη δημιουργία ενός ενιαίου περιβάλλοντος που περιλαμβάνει όλα τα απαραίτητα εργαλεία και βιβλιοθήκες για την εκτέλεση της εφαρμογής, χωρίς να απαιτείται εγκατάσταση Python ή άλλων εξαρτήσεων από τον τελικό χρήστη. Έτσι, η εφαρμογή μπορεί να τρέξει με συνέπεια σε οποιοδήποτε λειτουργικό σύστημα.

### Στοιχεία της dockerization

Η διαδικασία dockerization περιλάμβανε τα εξής βασικά αρχεία:

- **Dockerfile:** Ορίζει τις οδηγίες για το πώς χτίζεται η εικόνα (image). Περιλαμβάνει τη βάση (π.χ. python:3.12), την εγκατάσταση των εξαρτήσεων από `requirements.txt`, και την εκκίνηση του `app.py` με Streamlit.
- **.dockerignore:** Περιέχει λίστα αρχείων και φακέλων που αγνοούνται κατά την αντιγραφή στον container (π.χ. `.venv`, `.idea/`, `*.csv`, `*.pkl`), ώστε να μειωθεί το μέγεθος της εικόνας και να επιταχυνθεί η διαδικασία build.

### Χρήση σε Python (Pycharm)

Για την εκτέλεση του dockerization, ακολουθήθηκαν τα εξής βήματα:

- Εγκαταστάθηκε το Docker Desktop για Windows, με ενεργοποιημένη την υποστήριξη για WSL2 (Linux Subsystem).
- Δημιουργήθηκε ο φάκελος του project με όλα τα απαραίτητα αρχεία μέσα (app.py, Dockerfile, requirements, dockerignore,...)
- Από τον τερματικό του python, εκτελέστηκε η εντολή `docker build` για δημιουργία της εικόνας.
- Τέλος, η εφαρμογή ξεκίνησε με την εντολή `docker run`.

Η εφαρμογή εκκινεί αυτόματα μέσω Streamlit στη διεύθυνση `http://localhost:8501`, και μπορεί να χρησιμοποιηθεί από οποιοδήποτε browser.

## 5.1 Εκτέλεση Docker

Ο χρήσης θα πρέπει να κάνει clone (π.χ. με gitbash) η να κατεβάσει τα αρχεία από το github αποθετήριο και μετά να τρέξει στο τερματικό (σε φάκελο με τα αρχεία) τις εντολές:

```
docker build -t molecular-bio-app .
docker run --rm -p 8501:8501 molecular-bio-app
```

# 6 Πώς δημιουργήθηκε η αναφορά σε LaTeX

Η αναφορά συντάχθηκε σε περιβάλλον Overleaf με χρήση της γλώσσας LaTeX, προσαρμοσμένη για την ελληνική γλώσσα.

Για να υποστηριχθούν ελληνικά:

- Χρησιμοποιήθηκε το πακέτο `polyglossia` με επιλογή της γλώσσας `greek`.
- Ορίστηκε κατάλληλη γραμματοσειρά με χρήση `fontspec` (π.χ. Times New Roman).
- Η μεταγλώττιση (compilation) έγινε με χρήση του `LuaLaTeX`, που υποστηρίζει unicode χαρακτήρες και πολυτονικά.

Οι εικόνες ενσωματώθηκαν μέσω του πακέτου `graphicx` και όλες οι εντολές LaTeX οργανώθηκαν σε κεφάλαια σύμφωνα με τη μορφή επιστημονικής εργασίας.

# 7 Δομή Αποθετηρίου GitHub

Το αποθετήριο περιέχει τους εξής φακέλους και αρχεία:

- `app.py` – κύριος κώδικας της εφαρμογής
- `Dockerfile` – οδηγίες για το χτίσιμο του container
- `requirements.txt` – λίστα με τα πακέτα Python που απαιτούνται
- `.dockerignore` – εξαιρεί άχρηστα αρχεία από την εικόνα Docker
- `.streamlit/config.toml` – ρυθμίσεις εμφάνισης της Streamlit, συμπεριλαμβανομένου custom χρωματικού θέματος
- `iris.csv` – παράδειγμα dataset με δεδομένα για εκπαίδευση/δοκιμή
- `saved-model.pkl` – προ-εκπαιδευμένο μοντέλο σε μορφή joblib
- `screenshots/` – screenshots της εφαρμογής για την αναφορά
- `report/` – πηγαίος κώδικας της LaTeX αναφοράς
- `README.md` – βασικές οδηγίες χρήσης και εγκατάστασης

Η πλήρης εφαρμογή είναι διαθέσιμη στη διεύθυνση:
github.com/Citypop15/inf2022176$_{S}treamlit$