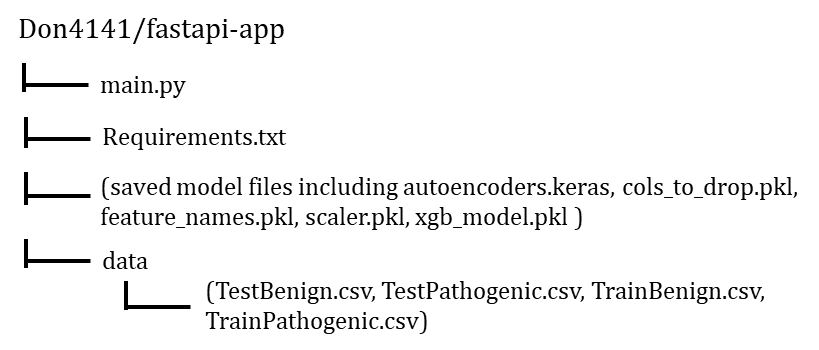
Steps followed to deploy FastAPI application on Render and have it accessible via the public URL

1. Train the model on your local machine to generate the necessary saved model files
2. Github Repository and code organization

* Organize the code files including the FastAPI app named main.py, requirement.txt (file contains all required dependencies), saved models, and datasets.

1. Push to Github

* File structure in the Github repository



* Commit and push the FastAPI code named main.py, the requirement.txt and all other saved model files as listed above to the Github repository

1. Set up a web service on Render to create the FastAPI backend

* Create an account or log in at <https://render.com>
* In Render dashboard, click “New” and choose “Web service” to create a new web service
* Link the GitHub account and select the repository containing the FastAPI app (fastapi-app/main.py)
* Configure the Service by giving it a descriptive name (Clinical Relevance of Missense Mutations Prediction API)

1. Step up deploy commands

* Step the start command to “uvicorn main:app --host 0.0.0.0 --port $PORT`”
* Configure instance type/region by choosing the appropriate instance type and region for the app

1. Deploy the service

* Click on create the service. Render will pull the repository, install the dependencies in the requirements.txt file, and deploy the Fastapi service.

1. Monitor application logs

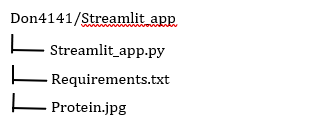
* Use Render’s dashboard to monitor build and runtime logs for troubleshooting

Once deployed, Render provided a public URL ([https://predicting-clinical-relevance-of.onrender.com](https://predicting-clinical-relevance-of.onrender.com/docs)). Test the API using endpoints “[https://predicting-clinical-relevance-of.onrender.com/docs#](https://predicting-clinical-relevance-of.onrender.com/docs)” for the interactive Swagger UI.

Steps followed to deploy streamlit application on Streamlit Community Cloud and have it accessible via the public URL

1. Github Repository and code organization

* Organize the code files including the streamlit app named streamlit\_app.py, Protein.jpg and requirement.txt (create this file which will contain all the required dependencies)
* File structure in the Github repository



* Commit and push the Streamlit app code named streamlit\_app.py (update script with the public URL provided by Render), Protein.jpg, and the requirement.txt from your local machine (was generated on Ubuntu) to the Github repository

2) Set up the Streamlit App on Streamlit Community Cloud to create the Streamlit frontend

* Sign up or log in at <https://share.streamlit.io>
* In streamlit dashboard, click “My new app”
* Select the Github repository and folder containing the streamlit script (Streamlit-app/streamlit\_app.py)
* Choose a descriptive name to the app URL (<https://missense-variant-prediction.streamlit.app/>)

3) Deploy the service

* Click “Deploy” and wait for the build process to complete
* Once the build was successful, Streamlit Cloud provided a public URL that can be used to view and interact with the app

4) Testing and Interaction

* Open the public URL in a browser to verify that the app is live and functioning as expected