DESIGN GUI FOR A COMPARATIVE STRUCTURE MODELING TOOL NAMED MODELLER

PROJECT SUBMITTED
IN PARTIAL FULFILMENT OF THE REQUIREMENT
FOR THE AWARD OF THE DEGREE
OF

BACHELOR OF ENGINEERING
IN
BIOTECHNOLOGY

BY

ABHINAV MATHUR

(BE/1016/06)

DEPARTMENT OF BIOTECHNOLOGY
BIRLA INSTITUTE OF TECHNOLOGY
MESRA, RANCHI – 835 215
(2009-2010)

DECLARATION CERTIFICATE

I hereby, certify that the work which has been presented in the thesis entitled "**Design GUI for a Comparative Structure Modeling Tool named MODELLER"** in partial fulfillment of the requirement for the award of the Degree of **Bachelor of Engineering**, submitted in the Department of Biotechnology, Birla Institute of Technology, Mesra, Ranchi is an authentic record of my work under the supervision of **Dr. A.S.Vidyarthi.**

The results embodied in this thesis have not been submitted by me or anybody else to any other University of Institute for the award of any Degree or Diploma.

Date: **(ABHINAV MATHUR)**

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Date **(Dr. A.S.VIDYARTHI)**
Dept. of Biotechnology
B.I.T., Mesra

Forwarded and Recommended for Submission.

Head,
Department of Biotechnology
Birla Institute of Technology, Mesra,
Ranchi-835215

Advisor (UG Projects)
Birla Institute of Technology, Mesra
Ranchi-835215

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to all those who helped me in this project. First of all I would like to thank my Professor and Supervisor Dr.A.S.Vidyarthi, Head, Department of Biotechnology who helped me on each stage throughout the project.

My special thanks to Mr. Ashutosh Kumar for giving me the idea of doing this project and to Mr. Shankaracharya and Mr. Santosh Kumar Jha for their continuous guidance during the development of this project.

Last but not the least my sister Ms. Shweta Mathur for extending technical as well as never ending moral support. Support from my family and friends at every stage of development of this project is priceless and my special vote of thanks goes to them for steering me through this project.

DATE : (**ABHINAV MATHUR**)

TABLE OF CONTENTS

INTRODUCTION

Functional characterization of a protein sequence is one of the most frequent problems in biology. This task is usually facilitated by an accurate three-dimensional (3-D) structure of the studied protein. In the absence of an experimentally determined structure, comparative or homology modeling often provides a useful 3-D model for a protein that is related to at least one known protein structure. Comparative modeling predicts the 3-D structure of a given protein sequence (target) based primarily on its alignment to one or more proteins of known structure (templates).The necessary conditions are that the similarity between them can be constructed. This approach to structure prediction is possible because a small change in the protein sequence usually results in a small change in its 3D structure.

Uses of Comparative Protein Structure Models

The 3D structure of a protein generally provides more information about its function than sequence because interactions of a protein with other molecules are determined by amino acid residues that are close in space but are frequently distant in sequence. Comparative modeling remains the only method that can reliably predict the 3D structure of a protein with accuracy comparable to that of low resolution structures. Typical uses of comparative modeling are:

- Designing (site- directed) mutants to test hypotheses about function
- Identifying active and binding sites.
- Searching for ligands in a given binding site.
- Designing and improving ligands of a given binding site.
- Modeling substrate specificity.
- Predicting antigenic epitopes.
- Protein-protein docking simulations.
- Testing a given sequence – structure alignment.
- Inferring function from calculated electrostatic potential around the protein

Steps in Comparative Modeling

Comparative modeling consists of five main steps:

(i) Search for templates,

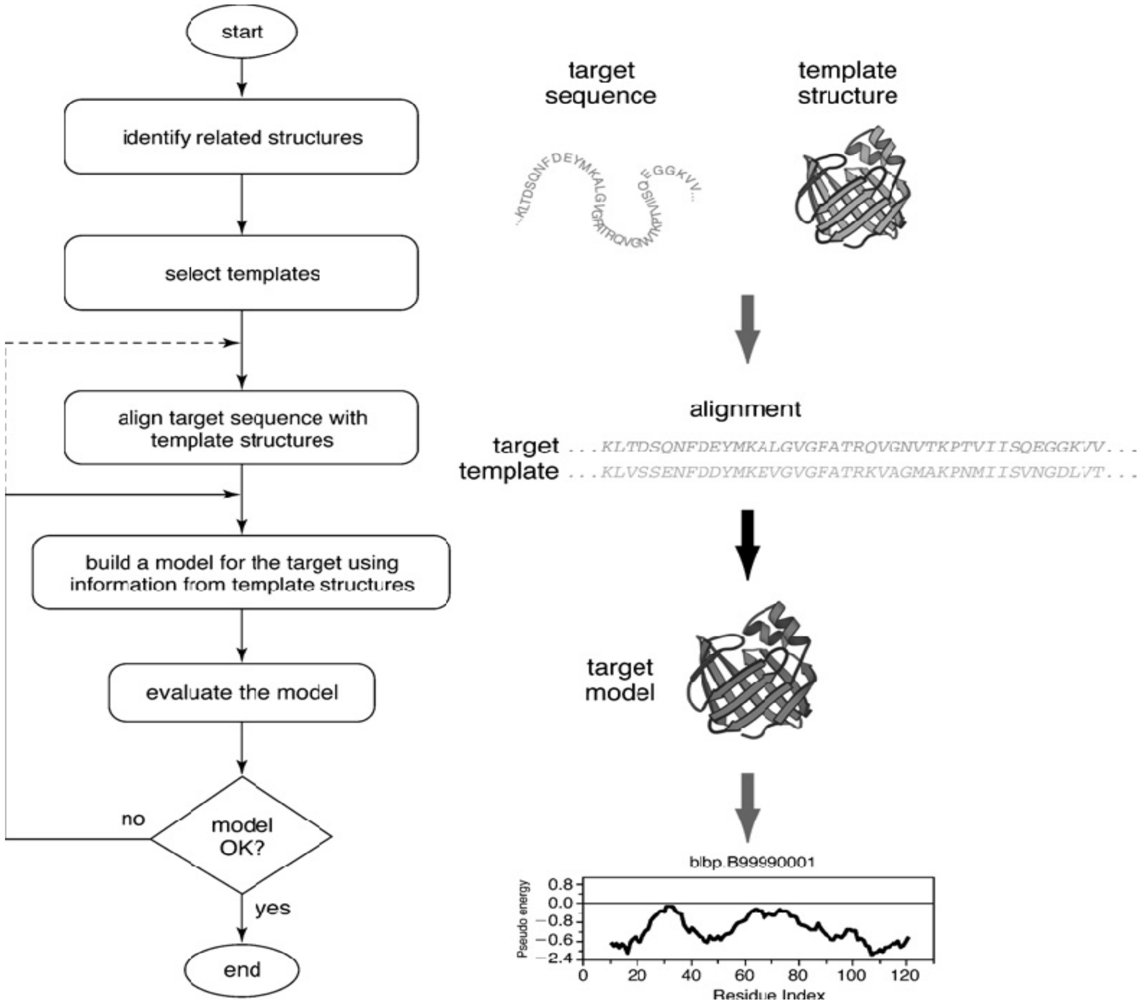
(ii) Selection of one or more templates,

(iii) Target-template alignment,

(iv) Model building

(v) Model evaluation

The steps in comparative modeling can be shown by the given diagram:


start
identify related structures
select templates
align target sequence with template structures
build a model for the target using information from template structures
evaluate the model
model OK?
no
yes
end
target sequence
...KLTDSQNFDEYMKALGVGFATRQVGNVTKPTVIISQEGGKVV...
template structure
alignment
target ...KLTDSQNFDEYMKALGVGFATRQVGNVTKPTVIISQEGGKVV...
template ...KLVSSENFDDYMKEVGVGFATRKVAGMAKPNMIISVNGDLVT...
target model
blbp.B99990001
Pseudo energy
0.8
0.0
−0.8
−0.6
−2.4
0
20
40
60
80
100
120
Residue Index

Comparative Modeling with Program “MODELLER”

MODELLER is a computer program for comparative protein structure modeling developed by **Andrej Sali** at the University of California, San Francisco. In the simplest case, the input is an alignment of a sequence to be modeled with the template structure(s), the atomic coordinates of the template(s), and a simple script file. MODELLER then automatically calculates a model containing all non-hydrogen atoms, without any user intervention and within minutes on a desktop computer. Apart from model building, MODELLER can perform auxiliary tasks such as fold-assignment alignment of two protein sequences or their profiles, multiple alignment of protein sequences and/or structures, clustering of sequences and/or structures, and ab initio modeling of loops in protein structures.

MODELLER implements comparative protein structure modeling by satisfaction of spatial restraints that include (i) homology-derived restraints on the distances and dihedral angles in the target sequence, extracted from its alignment with the template structures , (ii) stereochemical restraints such as bond length and bond angle preferences, (iii) statistical preferences for dihedral angles and non-bonded inter-atomic distances, obtained from a representative set of known protein structures and (iv) optional manually curated restraints, such as those from NMR spectroscopy, rules of secondary structure packing, cross-linking experiments, fluorescence spectroscopy, image reconstruction from electron microscopy, site-directed mutagenesis, and intuition The spatial restraints, expressed as probability density functions, are combined into an objective function that is optimized by a combination of conjugate gradients and molecular dynamics with simulated annealing. This model building procedure is similar to structure determination by NMR spectroscopy.

A typical operation in MODELLER would consist of (i) preparing an input Python script, (ii) ensuring that all required files (sequences, structures, alignments, etc.) exist, (iii) executing the input script by typing mod8v2 <input-script>, and (iv) analyzing the output and log files.

THE DATA FLOW IN MODELLER

The procedure for calculating a 3-dimensional model for a sequence with unknown structure using MODELLER is as follows:

1. Fold Assignment:

   - The first step in comparative modeling is to identify one or more template structure(s) that have detectable similarity to the target. This identification is achieved by scanning the sequence of unknown structure against a library of sequences extracted from known protein structures in the Protein Data Bank (PDB). This step is performed using the profile.build() module of MODELLER (file build_profile.py. The profile.build() command uses the local dynamic programming algorithm to identify related sequences. In the simplest case, profile.build() takes as input the target sequence (target.ali) and a database of sequences of known structure (file pdb_95.pir) and returns a set of statistically significant alignments (file build_profile.prf).

   - The results of the scan are stored in the output file called build_profile.prf. The first six lines of this file contain the input parameters used to create the alignments. Subsequent lines contain several columns of data; for the purposes of this example, the most important columns are (i) the second column, containing the PDB code of the related template sequences; (ii) the eleventh column, containing the percentage sequence identity between the target and template sequences; and (iii) the twelfth column, containing the E-values for the statistical significance of the alignments.

   - The extent of similarity between the target-template pairs is usually quantified using sequence identity or a statistical measure such as E-value. Inspection of column shows the template with the highest sequence identity with the target.

2. Sequence-Structure Alignment

- Sequence-structure alignments will be calculated using the align2d() module of MODELLER. Although align2d() is based on a global dynamic programming algorithm, it is different from standard sequence-sequence alignment methods because it takes into account structural information from the template when constructing an alignment. This task is achieved through a variable gap penalty function that tends to place gaps in solvent exposed and curved regions, outside secondary structure segments, and between two positions that are close in space.

- Single-Template
    - The input script align2d.py reads in the structure of the chosen template and the target sequence and calls the align2d() module to perform the alignment. The resulting alignment is written out to the specified alignment files in the PIR format and in the PAP format.
- Multiple-Template
    - The first step in using multiple templates for modeling is to obtain a multiple structure alignment of all the chosen templates. The structure alignment module of MODELLER, salign(), can be used for this purpose. The input script salign.py contains the necessary Python instructions to achieve a multiple structure alignment. The script reads in all the six template structures into an alignment object and then calls salign() to generate the multiple structure alignment.
    - The next step is to align the unknown sequence with the multiple structure alignment generated above. This task is accomplished using the script file align2d-mult.py, that again calls the align2d() module to calculate the sequence-structure alignment. Upon execution, the resulting alignments are written to in the PIR and PAP formats, respectively.

3. Model Building

- Two variations of the model building protocol will be described, corresponding to the two alignments generated above: (i) modeling using a single template and (ii) modeling using multiple templates, followed by building and optimizing a consensus model. The files required for each of these protocols are present in separate subdirectories called single/ and multiple/, respectively.

- Single Template
    - The input script model-single.py lists the Python commands necessary to build the model of the unknown sequence using the information derived from template structure. The script calls the automodel class specifying the name of the alignment file to use and the identifiers of the target and template sequences. The starting_model and ending_model specify the number of models that should be calculated by randomizing the initial coordinates. The models are then assessed with the GA341 and DOPE assessment functions.
    - Upon completion, all the models for the target sequence are written out in the PDB format.

- Multiple Templates with Consensus Modeling
    - The input script, model-mult.py, is quite similar to model-single.py. The specification of the template codes to automodel now contains all the chosen PDB codes and additionally, the cluster() method is called to exploit the diversity of the generated models via a clustering and optimization procedure to construct a single consensus model
    - Upon completion, the models for the target sequence and the consensus model are written in pdb format.

4. Model Evaluation

- The log files produced by each of the model building procedures (model-single.log and model-mult.log) contain a summary of each calculation at the bottom of the file. This summary includes, for each of the models, the MODELLER objective function the DOPE pseudo-energy value and the value of the GA341 score. These scores can be used to identify which of the models produced is likely to be the most accurate model (A residue-based pseudo-energy profile for the best scoring model, chosen as the one with the lowest DOPE statistical potential score, can be obtained by executing the evaluate_model.py script. Such a profile is useful to detect local regions of high pseudo-energy that usually correspond to errors in the model.

ADVANTAGES OF MODELLER

- It is an offline tool. So, it doesn’t require any internet connectivity.
- Performs and displays all steps of Homology Modeling.
- Allows selection of templates from the downloadable PIR database.
- Allows use of multiple templates.
- Provides information about the DOPE score and GA341 score of the models.
- Scope of improvement of the models so formed by features like Loop Modeling or by editing target-template alignment.
- No requirement of sharing data anywhere (as in the case of SWISS-MODEL server).

DRAWBACKS OF MODELLER

- Works on Command Line arguments.
- Lacks Graphic User Interface (GUI).
- Complex software protocol requiring extensive study of MODELLER manuals and tutorials.
- Knowledge of Python scripts required for advanced usage.
- Cumbersome visualization of output in form of verbose files.
- Consumes lot of time in writing scripts and formatting the inputs.

OBJECTIVES

- To design an interactive Graphic User Interface for MODELLER that reduces complexity and eases its use.
- To eliminate the cumbersome formatting of the inputs, scripting processes and analysis of verbose output files.
- To demonstrate all the steps of Homology Modeling, screening out the backend processes which require extensive study of manuals and tutorials.
- To encourage people draw inferences with help of MODELLER software without encountering its demerits of need of a sound bioinformatics as well as software knowledge.
- To use JAVA for application development in order to add the scope of making the application platform independent that is it will work on Windows, Macintosh OS and Linux.

METHODOLOGY

The screens are so designed that they portray the procedure and the flow of MODELLER GUI in a definite sequence.

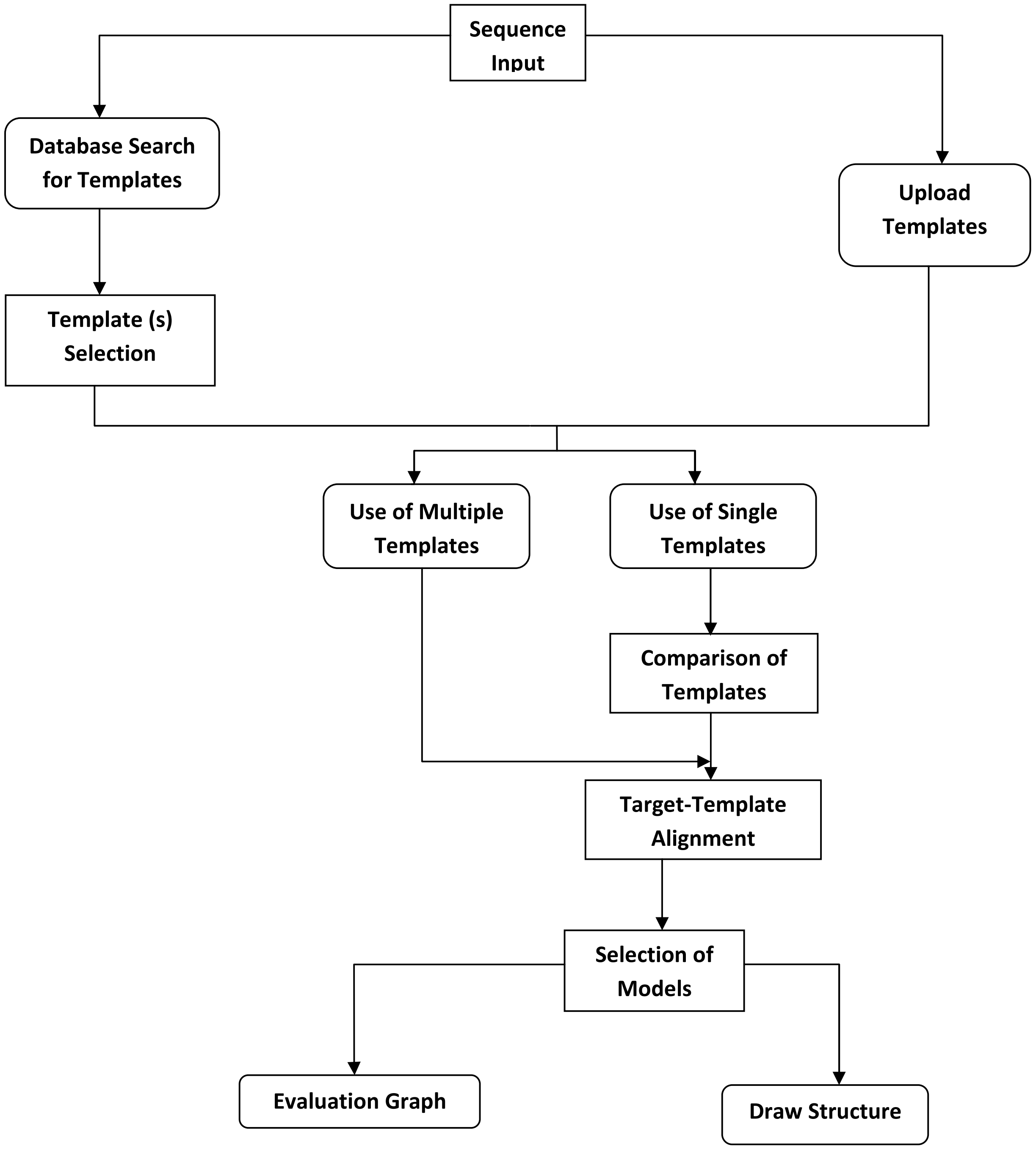


Fig.:- GUI Work Flow

The screens were developed using a development IDE NetBeans v6.7.1. It consists of a fully equipped palette with all the required machinery to develop an attractive Graphic User Interface. It also eases out the complexity of writing source code for the application through its extensive hints and help library.

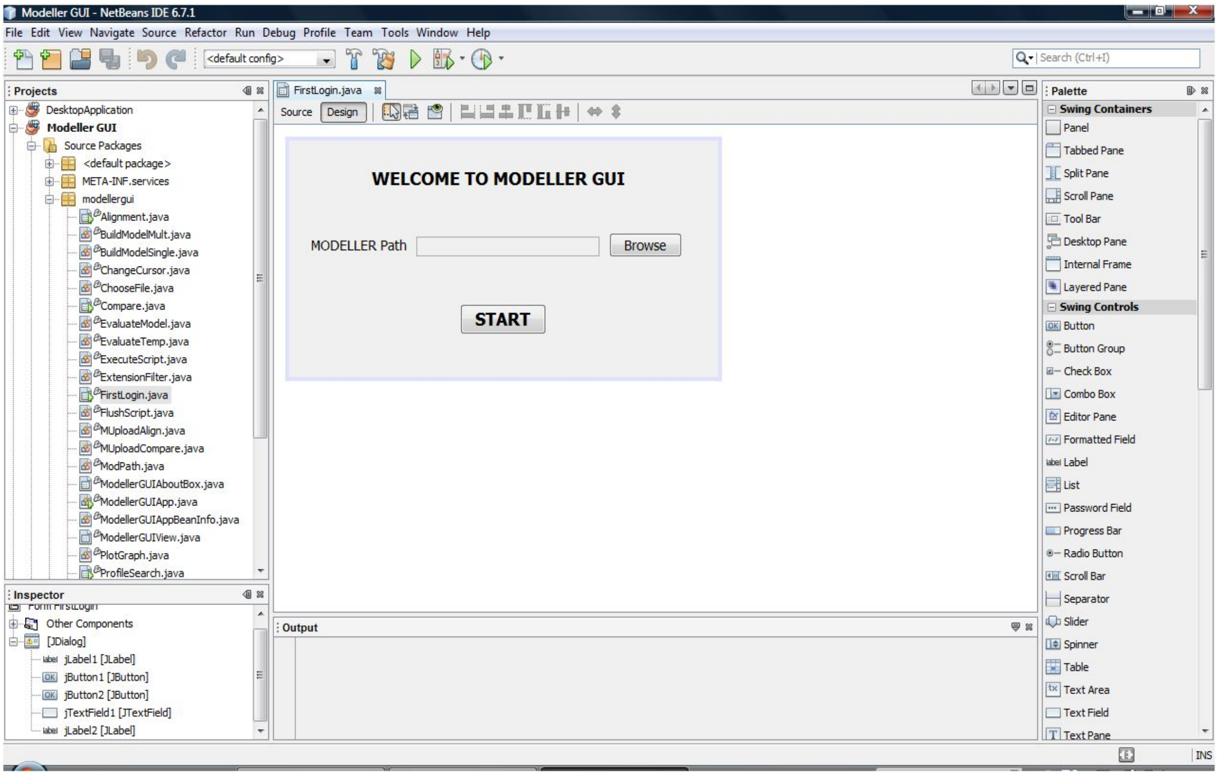
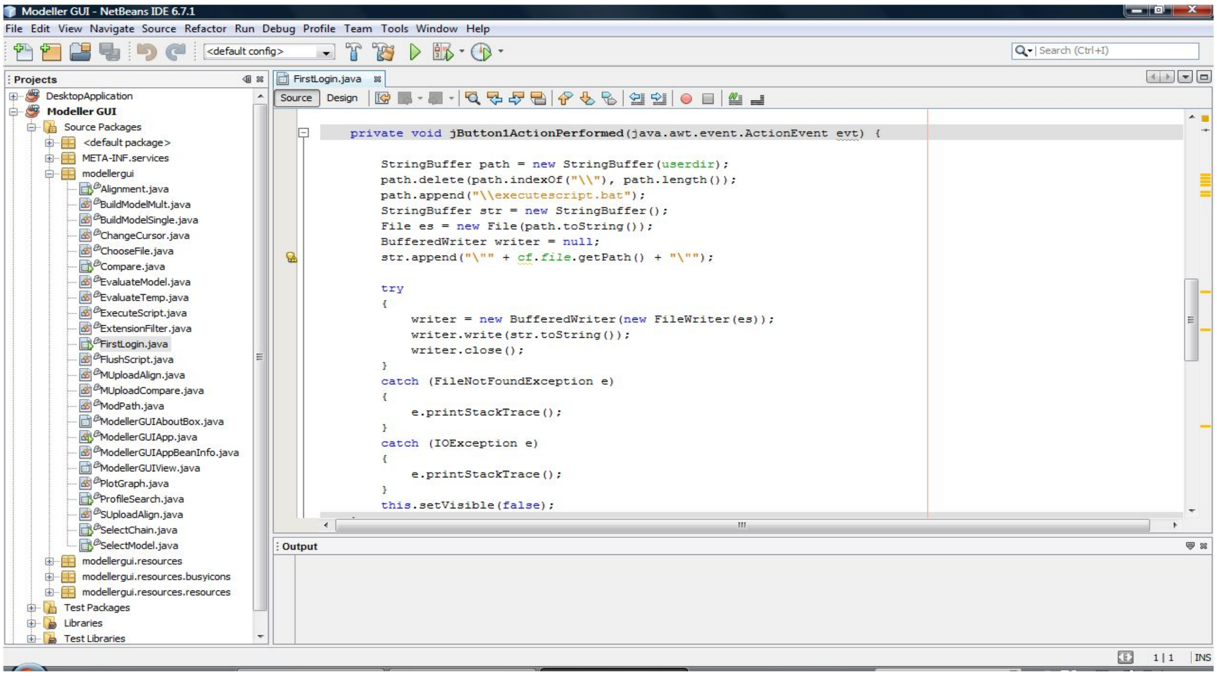


Fig. :- Screenshot of the NetBeans IDE 6.7.1 in design view (above) and source view (below)



```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

    StringBuffer path = new StringBuffer(userdir);
    path.delete(path.indexOf("\\"), path.length());
    path.append("\\executescript.bat");
    StringBuffer str = new StringBuffer();
    File es = new File(path.toString());
    BufferedWriter writer = null;
    str.append("\"" + cf.file.getPath() + "\"");

    try
    {
        writer = new BufferedWriter(new FileWriter(es));
        writer.write(str.toString());
        writer.close();
    }
    catch (FileNotFoundException e)
    {
        e.printStackTrace();
    }
    catch (IOException e)
    {
        e.printStackTrace();
    }
    this.setVisible(false);
```

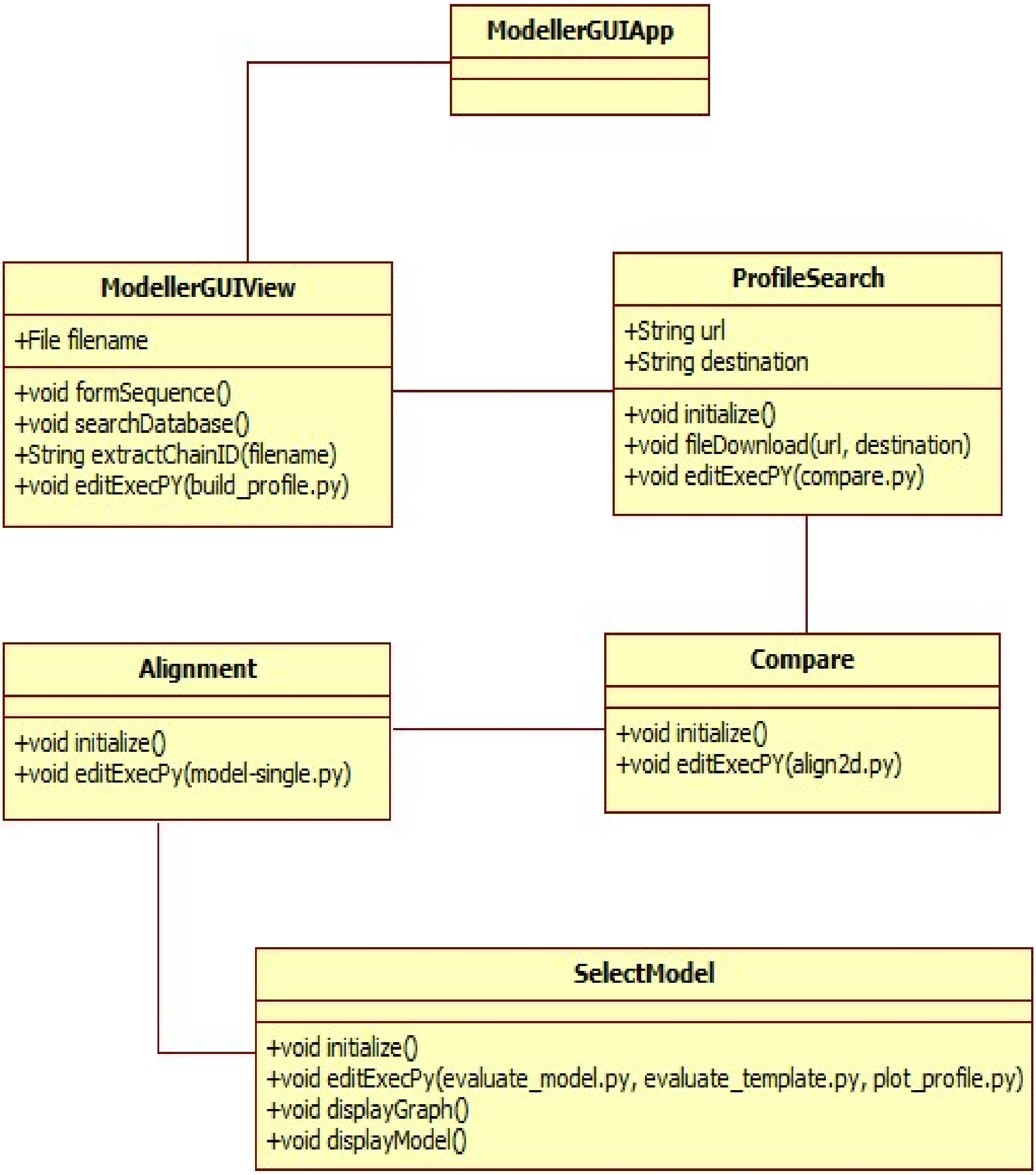


Fig.:- GUI Class Diagram with Inter-linking

All the screens designed in the software along with its user features and actions are summarized in the same sequence below in tabular form.

Screen Description	Component	Type	Action Performed	Functions Defined in Source Code
Sequence Input & Database Search/Upload Template(s)	Title of Project	Text Field	Formation of MODELLER recognizable .ali file	formSequence()
	Sequence	Text Area		
	Clear	Label	Clears the sequence area	
	Database Search for Templates	Radio Button	Notifies the GUI to search for templates in the downloaded PIR database	searchDatabase()
	Upload the Template	Radio Button	Gives the option of browsing for file	
	Select Template(s) & Chain(s)	Label	Invokes the pop-up window for selection of chain(s)	extractChainID()
	Use Multiple Templates	Checkbox	Notifies the GUI to use multiple templates for Target-Template alignment	
	Go	Button	Edits the python scripts build_profile.py / compare.py / salign.py & align2d_mult.py as per the decision made by the user. Execution of the above mentioned scripts and extraction of the required data from output files which is displayed in	editExecPY (build_profile.py)

			the next screen	
Selection of Template(s) & Chain(s) (Pop-up Window)	Uploaded Templates List	Table	Allows final selection of the uploaded template(s) and displays extracted chain(s) from the template PDB file(s)	
	Go	Button	Saves the selected templates and chains in a text file for future use	
Database Search Results	Database Search Results (Templates)	Text Area	Indicates the input parameters used in MODELLER to build the profile	initialize()
		Table	Displays the templates with sequence similarity to the target and their E-value	
	Select All	Button	Selects all the templates	
	Reset	Button	Deselects all checkboxes	
	Download	Button	Downloads template files from PDB website (requires internet connection)	fileDownload (url, destination)
	Use Multiple Templates	Checkbox	Notifies the GUI to use multiple templates for Target-Template alignment	
	Next	Button	Edits the python scripts compare.py / salign.py & align2d_mult.py as per the decision made by the	editExecPY (compare.py)

			user. Execution of the above mentioned scripts and extraction of the required data from output files which is displayed in the next screen	
Template Comparison	Dendrogram of Template Comparison	Text Area	Displays dendrogram of all selected templates with their Crystallographic resolution factor and phylogenetic score	initialize()
	Selection of Template	Table	Allows selection of the best template	
	Next	Button	Edits the python scripts align2d.py. Execution of the above mentioned script and extraction of the required data from output files which is displayed in the next screen	editExecPY (align2d.py)
Target-Template Alignment	Template – Target Alignment	Text Area	Displays the alignment between the target sequence and the template structure(s)	initialize()
	Edit	Label	Allows the user to edit the Target-Template alignment (presently inactive feature)	

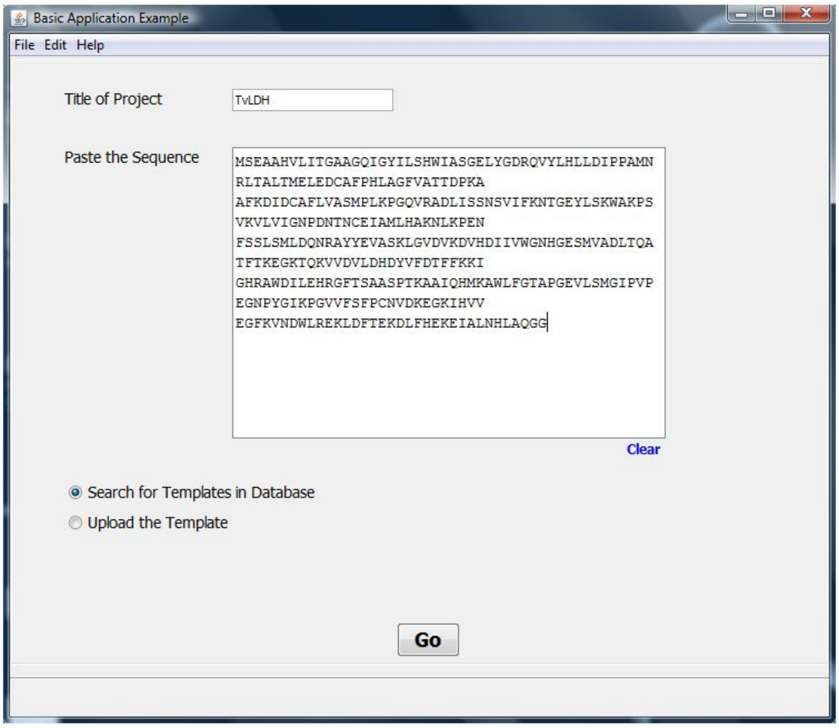
	Number of Models	Text Field	Inputs the number of models to be made	
	Next	Button	Edits the python scripts model-single.py / model_mult.py as per the decision made by the user. Execution of the above mentioned scripts and extraction of the required data from output files which is displayed in the next screen	editExecPY (model-single.py)
Built Models Assessment Score	Assessment Score of Models	Table	Allows the selection of suitable model based on DOPE and GA341 score	initialize()
	Plot Evaluation Graph / Draw Structure	Button	Edits the python scripts evaluate_model.py, evaluate_template.py and plot_profiles.py. Execution of the above mentioned scripts and opens the evaluation graph image (in default Windows program) and the PDB file (using JMol) for visualization	editExecPY (evaluate_model.py, evaluate_template.py, plot_profiles.py), drawGraph(), drawModel()

RESULTS AND DISCUSSION

The GUI is demonstrated by taking an example modeling of lactate dehydrogenase from *Trichomonas vaginalis*. A novel gene for lactate dehydrogenase was identified from the genomic sequence of *Trichomonas vaginalis* (TvLDH). The corresponding protein had a higher similarity to the malate dehydrogenase of the same species (TvMDH) than to any other LDH.
The individual modeling steps of this example are explained below.

A. Searching for Structures Related to TvLDH

In the first screen, the model file name (TvLDH) and the lactate dehydrogenase sequence were taken as input in the first two text fields respectively .The option of search for templates in Database was selected which searches for template structures with appropriate percentage identity with the target sequence using the build_profile.py script at the backend.



Title of Project: TvLDH

Paste the Sequence:

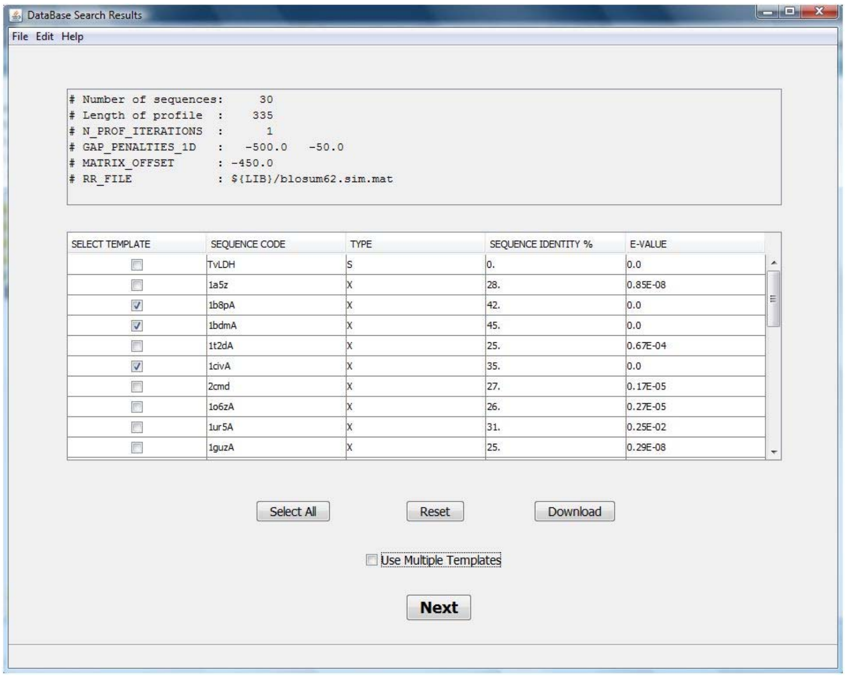
```
MSEAAHVLITGAAGQIGYILSHWIASGELYGDRQVYLHLLDIPPAMN
RLTALTMELEDCAFPHLAGFVATTDPKA
AFKDIDCAFLVASMPLKPGQVRADLISSNSVIFKNTGEYLSKWAKPS
VKVLVIGNPDNTNCEIAMLHAKNLKPEN
FSSLSMLDQNRAYYEVASKLGVDVKDVHDIIVWGNHGESMVADLTQA
TFTKEGKTQKVVDVLDHDYVFDTFFKKI
GHRAWDILEHRGFTSAASPTKAAIQHMKAWLFGTAPGEVLSMGIPVP
EGNPYGIKPGVVFSFPCNVDKEGKIHVV
EGFKVNDWLREKLDFTEKDLFHEKEIALNHLAQGG
```

● Search for Templates in Database
○ Upload the Template

Fig.:- Initial screen to input sequence and title

B. **Selection and Downloading of Templates**

The output of the "build_profile.py" script was written to the "build_profile.log" file. MODELLER writes the profile in text format to the "build_profile.prf" file. An extract of the output file can be seen in next screen. The first 6 commented lines indicate the input parameters used in MODELLER to build the profile. Subsequent lines correspond to the detected similarities by profile.build(). In general, a sequence identity value above approximately 25% indicates a potential template unless the alignment is short (i.e., less than 100 residues). A better measure of the significance of the alignment is given by the e-value of the alignment. Six PDB sequences showed very significant similarities to the query sequence with e-values equal to 0 (1bdm:A, 5mdh:A, 1b8p:A, 1civ:A, 7mdh:A, and 1smk:A). The selected template(s) were downloaded in .gz format from PDB website (ftp://ftp.wwpdb.org/) and extracted into the working folder in .pdb format.



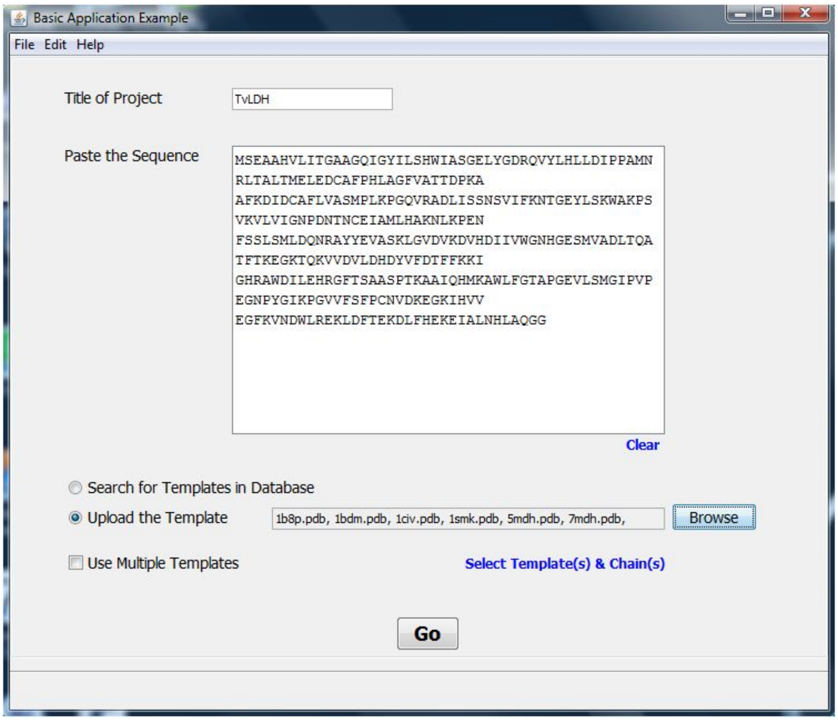
```
# Number of sequences:     30
# Length of profile  :    335
# N_PROF_ITERATIONS  :      1
# GAP_PENALTIES_1D   :   -500.0   -50.0
# MATRIX_OFFSET      : -450.0
# RR_FILE            : ${LIB}/blosum62.sim.mat
```

SELECT TEMPLATE	SEQUENCE CODE	TYPE	SEQUENCE IDENTITY %	E-VALUE
☐	TvLDH	S	0.	0.0
☐	1a5z	X	28.	0.85E-08
☑	1b8pA	X	42.	0.0
☑	1bdmA	X	45.	0.0
☐	1t2dA	X	25.	0.67E-04
☑	1civA	X	35.	0.0
☐	2cmd	X	27.	0.17E-05
☐	1o6zA	X	26.	0.27E-05
☐	1ur5A	X	31.	0.25E-02
☐	1guzA	X	25.	0.29E-08

Fig.:- Database search results

C. **Uploading Templates**

On selecting the upload the template option in the initial screen, user-defined template(s) were selected for modeling the target protein sequence. Selection of chains (extracted from template PDB files) was done in the popup screen made visible by clicking on the Select Template(s) & Chain(s).



Title of Project: TvLDH

Paste the Sequence:

```
MSEAAHVLITGAAGQIGYILSHWIASGELYGDRQVYLHLLDIPPAMN
RLTALTMELEDCAFPHLAGFVATTDPKA
AFKDIDCAFLVASMPLKPGQVRADLISSNSVIFKNTGEYLSKWAKPS
VKVLVIGNPDNTNCEIAMLHAKNLKPEN
FSSLSMLDQNRAYYEVASKLGVDVKDVHDIIVWGNHGESMVADLTQA
TFTKEGKTQKVVDVLDHDYVFDTFFKKI
GHRAWDILEHRGFTSAASPTKAAIQHMKAWLFGTAPGEVLSMGIPVP
EGNPYGIKPGVVFSFPCNVDKEGKIHVV
EGFKVNDWLREKLDFTEKDLFHEKEIALNHLAQGG
```

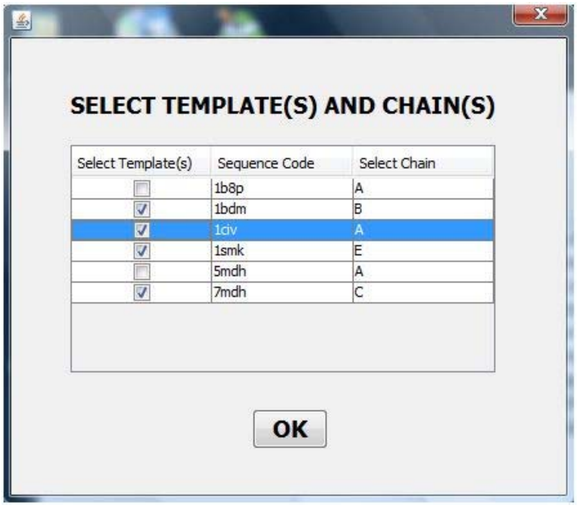
○ Search for Templates in Database

● Upload the Template: 1b8p.pdb, 1bdm.pdb, 1civ.pdb, 1smk.pdb, 5mdh.pdb, 7mdh.pdb,

☐ Use Multiple Templates

Select Template(s) & Chain(s)

Fig.:- (above) uploading template(s) and (below) selecting final template(s) and chain(s).

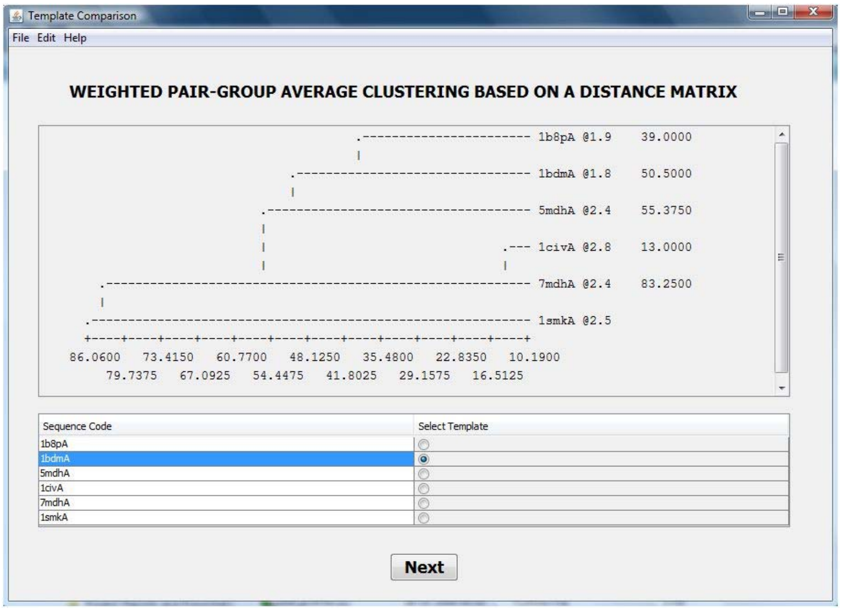


SELECT TEMPLATE(S) AND CHAIN(S)

Select Template(s)	Sequence Code	Select Chain
☐	1b8p	A
☑	1bdm	B
☑	1civ	A
☑	1smk	E
☐	5mdh	A
☑	7mdh	C

D. **Comparison of Templates**

To select the most appropriate template for our query sequence over the six similar structures, the alignment.compare_structures() command was used to assess the structural and sequence similarity between the possible templates. A file (compare.log) was written with pairwise sequence distances that can be used directly as the input to the dendogram which calculates a clustering tree from the input matrix of pairwise distances, which helps visualizing differences among the template candidates. Excerpts from the log file are shown in screen.



WEIGHTED PAIR-GROUP AVERAGE CLUSTERING BASED ON A DISTANCE MATRIX

```
                                       .----------------------- 1b8pA @1.9    39.0000
                                       |
                              .-------------------------------- 1bdmA @1.8    50.5000
                              |
                          .------------------------------------ 5mdhA @2.4    55.3750
                          |
                          |                                .--- 1civA @2.8    13.0000
                          |                                |
    .---------------------------------------------------------- 7mdhA @2.4    83.2500
    |
  .------------------------------------------------------------ 1smkA @2.5
  +----+----+----+----+----+----+----+----+----+----+----+----+
86.0600   73.4150   60.7700   48.1250   35.4800   22.8350   10.1900
     79.7375   67.0925   54.4475   41.8025   29.1575   16.5125
```

Sequence Code	Select Template
1b8pA	○
1bdmA	●
5mdhA	○
1civA	○
7mdhA	○
1smkA	○

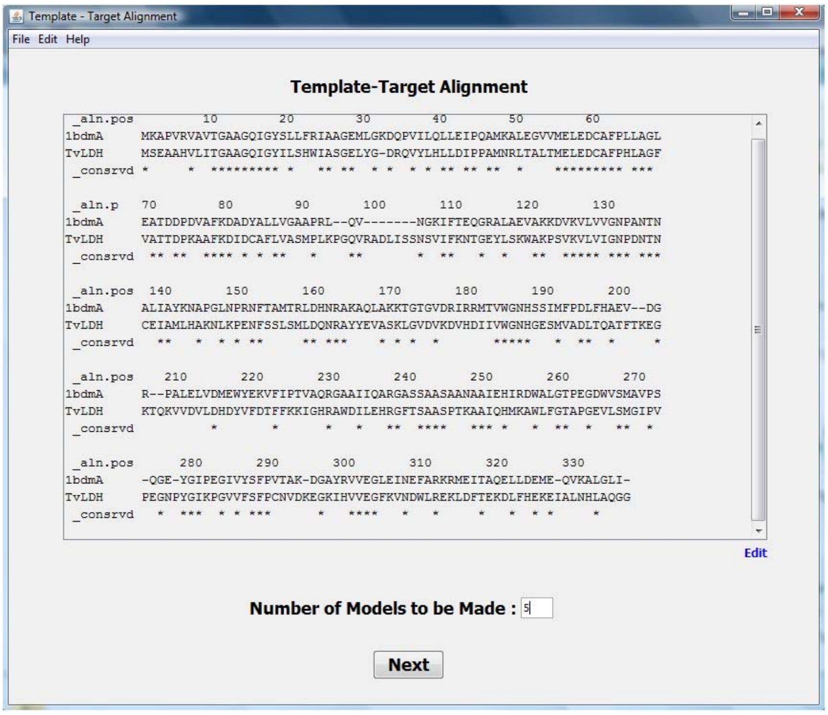
Fig. Dendogram showing crystallographic R-factor and phylogeny score

The comparison above shows that 1civ:A and 7mdh:A are almost identical, both sequentially and structurally. However, 7mdh:A has a better crystallographic resolution (2.4Å versus 2.8Å), eliminating 1civ:A. A second group of structures (5mdh:A, 1bdm:A, and 1b8p:A) share some similarities. From this group, 5mdh:A has the poorest resolution leaving for consideration only 1bdm:A and 1b8p:A. 1smk:A is the most diverse structure of the whole set of possible templates. However, it is the one with the lowest sequence identity (34%) to the query sequence. We finally picked 1bdm:A over 1b8p:A and

7mdh:A because of its better crystallographic R-factor (16.9%) and higher overall sequence identity to the query sequence (45%).

E. Target-Template Alignment

The MODELLER script, align2d.py, aligned the TvLDH sequence in file "TvLDH.ali" with the 1bdm:A structure in the PDB file "1bdm.pdb". The alignment was written out in two formats, PIR ("TvLDH-1bdmA.ali") and PAP ("TvLDH-1bdmA.pap"). The PIR format was used by MODELLER in the subsequent model building stage, while the PAP alignment format was easier to inspect visually. Due to the high target-template similarity, there were only a few gaps in the alignment. In the PAP format, all identical positions are marked with a "*".



Template-Target Alignment

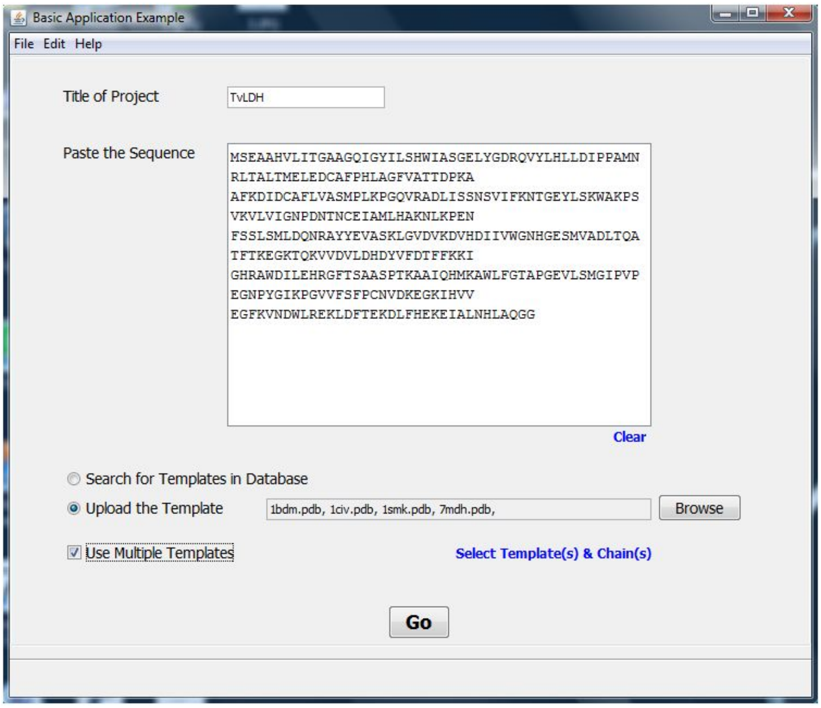
```
 _aln.pos         10        20        30        40        50        60
1bdmA     MKAPVRVAVTGAAGQIGYSLLFRIAAGEMLGKDQPVILQLLEIPQAMKALEGVVMELEDCAFPLLAGL
TvLDH     MSEAAHVLITGAAGQIGYILSHWIASGELYG-DRQVYLHLLDIPPAMNRLTALTMELEDCAFPHLAGF
 _consrvd *     *  ********* *   ** **  * *  * * ** ** **  *    ********* ***

 _aln.p   70        80        90       100       110       120       130
1bdmA     EATDDPDVAFKDADYALLVGAAPRL--QV-------NGKIFTEQGRALAEVAKKDVKVLVVGNPANTN
TvLDH     VATTDPKAAFKDIDCAFLVASMPLKPGQVRADLISSNSVIFKNTGEYLSKWAKPSVKVLVIGNPDNTN
 _consrvd  ** **  **** * * **   *    **       *  **   *  *   **  ***** *** ***

 _aln.pos  140       150       160       170       180       190       200
1bdmA     ALIAYKNAPGLNPRNFTAMTRLDHNRAKAQLAKKTGTGVDRIRRMTVWGNHSSIMFPDLFHAEV--DG
TvLDH     CEIAMLHAKNLKPENFSSLSMLDQNRAYYEVASKLGVDVKDVHDIIVWGNHGESMVADLTQATFTKEG
 _consrvd   **   *  * * **     ** ***    * * *  *       *****   *  **  *     *

 _aln.pos    210       220       230       240       250       260       270
1bdmA     R--PALELVDMEWYEKVFIPTVAQRGAAIIQARGASSAASAANAAIEHIRDWALGTPEGDWVSMAVPS
TvLDH     KTQKVVDVLDHDYVFDTFFKKIGHRAWDILEHRGFTSAASPTKAAIQHMKAWLFGTAPGEVLSMGIPV
 _consrvd          *       *      *   *   **  ****   *** *   *  **  *   **  *

 _aln.pos      280       290       300       310       320       330
1bdmA     -QGE-YGIPEGIVYSFPVTAK-DGAYRVVEGLEINEFARKRMEITAQELLDEME-QVKALGLI-
TvLDH     PEGNPYGIKPGVVFSFPCNVDKEGKIHVVEGFKVNDWLREKLDFTEKDLFHEKEIALNHLAQGG
 _consrvd   *  ***  * * ***      *   ****   *   *     *   *  * *     *
```

Number of Models to be Made : 5

Fig.:- Target-Template alignment and to input no. of models to be formed

F. Modeling with Multiple Templates

Differences in specificity between two similar proteins are depicted by precise and accurate models. Multiple templates are used to increase the accuracy of the models. Multiple templates option can be selected in both initial screen and the database search screen.



Title of Project: TvLDH

Paste the Sequence:

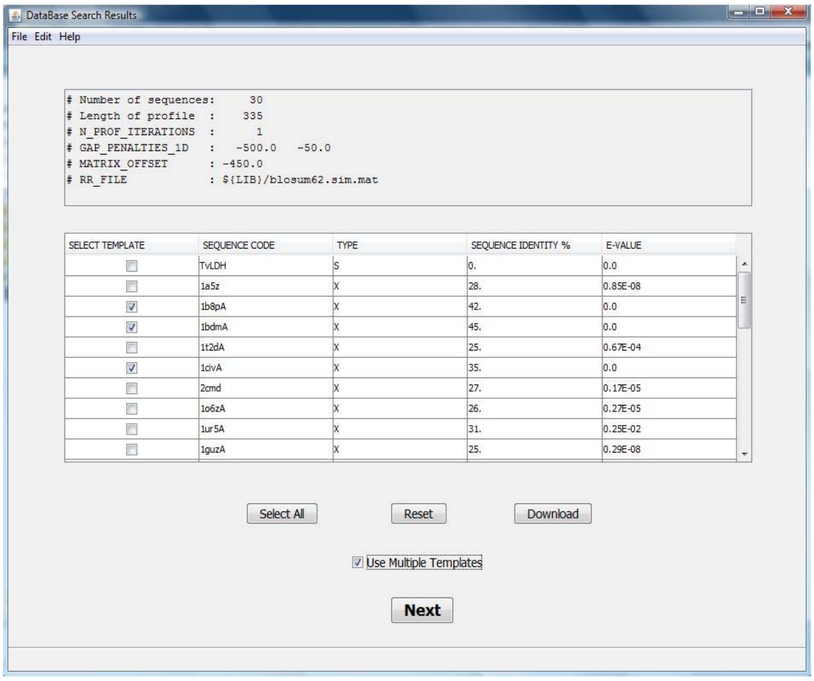
```
MSEAAHVLITGAAGQIGYILSHWIASGELYGDRQVYLHLLDIPPAMN
RLTALTMELEDCAFPHLAGFVATTDPKA
AFKDIDCAFLVASMPLKPGQVRADLISSNSVIFKNTGEYLSKWAKPS
VKVLVIGNPDNTNCEIAMLHAKNLKPEN
FSSLSMLDQNRAYYEVASKLGVDVKDVHDIIVWGNHGESMVADLTQA
TFTKEGKTQKVVDVLDHDYVFDTFFKKI
GHRAWDILEHRGFTSAASPTKAAIQHMKAWLFGTAPGEVLSMGIPVP
EGNPYGIKPGVVFSFPCNVDKEGKIHVV
EGFKVNDWLREKLDFTEKDLFHEKEIALNHLAQGG
```

○ Search for Templates in Database

● Upload the Template: 1bdm.pdb, 1civ.pdb, 1smk.pdb, 7mdh.pdb,

☑ Use Multiple Templates

Fig.:- Selection of multiple templates in initial screen (above) and Database search screen(below).

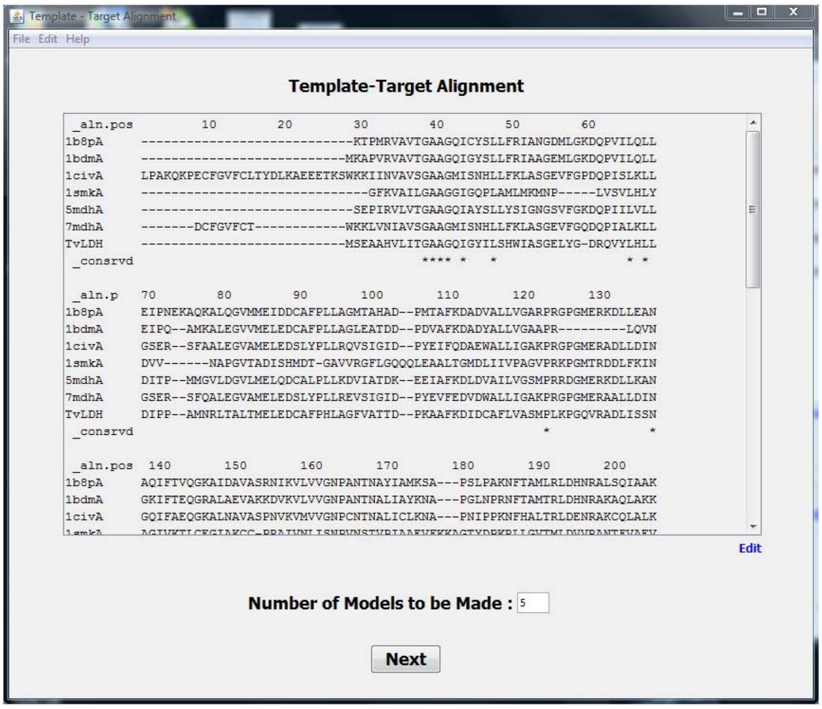


```
# Number of sequences:     30
# Length of profile  :    335
# N_PROF_ITERATIONS  :      1
# GAP_PENALTIES_1D   :   -500.0   -50.0
# MATRIX_OFFSET      : -450.0
# RR_FILE            : ${LIB}/blosum62.sim.mat
```

SELECT TEMPLATE	SEQUENCE CODE	TYPE	SEQUENCE IDENTITY %	E-VALUE
☐	TvLDH	S	0.	0.0
☐	1a5z	X	28.	0.85E-08
☑	1b8pA	X	42.	0.0
☑	1bdmA	X	45.	0.0
☐	1t2dA	X	25.	0.67E-04
☑	1civA	X	35.	0.0
☐	2cmd	X	27.	0.17E-05
☐	1o6zA	X	26.	0.27E-05
☐	1ur5A	X	31.	0.25E-02
☐	1guzA	X	25.	0.29E-08

☑ Use Multiple Templates

The multiple alignment was generated by the command salign() in MODELLER. All of the sequences are read from PDB files (using the append_model command), and then salign was used multiple times, to generate an initial rough alignment and then improve upon it by using more information. Next the query sequence was aligned to template structures. The alignment was then written out in both PIR and PAP formats which was visualised as follow.



Template-Target Alignment

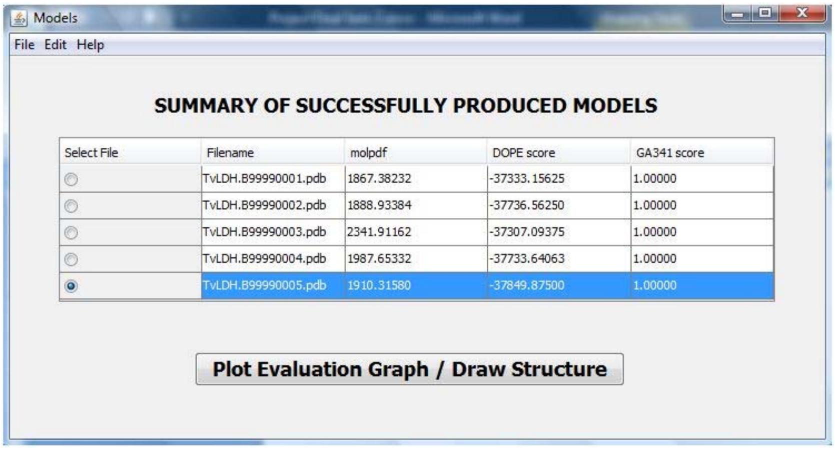
```
 _aln.pos          10        20        30        40        50        60
1b8pA      ----------------------------KTPMRVAVTGAAGQICYSLLFRIANGDMLGKDQPVILQLL
1bdmA      ---------------------------MKAPVRVAVTGAAGQIGYSLLFRIAAGEMLGKDQPVILQLL
1civA      LPAKQKPECFGVFCLTYDLKAEEETKSWKKIINVAVSGAAGMISNHLLFKLASGEVFGPDQPISLKLL
1smkA      ------------------------------GFKVAILGAAGGIGQPLAMLMKMNP-----LVSVLHLY
5mdhA      ----------------------------SEPIRVLVTGAAGQIAYSLLYSIGNGSVFGKDQPIILVLL
7mdhA      -------DCFGVFCT------------WKKLVNIAVSGAAGMISNHLLFKLASGEVFGQDQPIALKLL
TvLDH      ---------------------------MSEAAHVLITGAAGQIGYILSHWIASGELYG-DRQVYLHLL
 _consrvd                                      **** *   *                 * *

 _aln.p    70        80        90       100       110       120       130
1b8pA      EIPNEKAQKALQGVMMEIDDCAFPLLAGMTAHAD--PMTAFKDADVALLVGARPRGPGMERKDLLEAN
1bdmA      EIPQ--AMKALEGVVMELEDCAFPLLAGLEATDD--PDVAFKDADYALLVGAAPR---------LQVN
1civA      GSER--SFAALEGVAMELEDSLYPLLRQVSIGID--PYEIFQDAEWALLIGAKPRGPGMERADLLDIN
1smkA      DVV------NAPGVTADISHMDT-GAVVRGFLGQQQLEAALTGMDLIIVPAGVPRKPGMTRDDLFKIN
5mdhA      DITP--MMGVLDGVLMELQDCALPLLKDVIATDK--EEIAFKDLDVAILVGSMPRRDGMERKDLLKAN
7mdhA      GSER--SFQALEGVAMELEDSLYPLLREVSIGID--PYEVFEDVDWALLIGAKPRGPGMERAALLDIN
TvLDH      DIPP--AMNRLTALTMELEDCAFPHLAGFVATTD--PKAAFKDIDCAFLVASMPLKPGQVRADLISSN
 _consrvd                                                       *             *

 _aln.pos   140       150       160       170       180       190       200
1b8pA      AQIFTVQGKAIDAVASRNIKVLVVGNPANTNAYIAMKSA---PSLPAKNFTAMLRLDHNRALSQIAAK
1bdmA      GKIFTEQGRALAEVAKKDVKVLVVGNPANTNALIAYKNA---PGLNPRNFTAMTRLDHNRAKAQLAKK
1civA      GQIFAEQGKALNAVASPNVKVMVVGNPCNTNALICLKNA---PNIPPKNFHALTRLDENRAKCQLALK
1smkA      AGIVKTLCEGIAKCC-PPAIVNLISNPVNSTVPIAAEVFKKAGTYDPKRLLGVTMLDVVRANTFVAEV
```

Number of Models to be Made : 5

Fig.:- Template-Target Alignment

G. Building of Models

Once a target-template alignment was constructed, the model-single.py (or model_mult.py depending upon the selection of single or multiple template) script generated five similar models of TvLDH based on the 1bdm:A template structure and the alignment in file "TvLDH-1bdmA.ali" (TvLDH-mult.ali). The output file, "model-single.log"(model_mult.log) gave a summary of all the models built. For each model, it listed the file name, which contains the coordinates of the model in PDB format. The log also showed the score(s) of each model. The model with the lowest value of the MODELLER objective function or the DOPE assessment score or with the highest GA341 assessment score was picked. The molpdf and DOPE scores are not 'absolute' measures, in the sense that they can only be used to rank models calculated from the same alignment. Other scores are transferable.



SUMMARY OF SUCCESSFULLY PRODUCED MODELS

Select File	Filename	molpdf	DOPE score	GA341 score
○	TvLDH.B99990001.pdb	1867.38232	-37333.15625	1.00000
○	TvLDH.B99990002.pdb	1888.93384	-37736.56250	1.00000
○	TvLDH.B99990003.pdb	2341.91162	-37307.09375	1.00000
○	TvLDH.B99990004.pdb	1987.65332	-37733.64063	1.00000
●	TvLDH.B99990005.pdb	1910.31580	-37849.87500	1.00000

Fig.:- List of built models with DOPE and GA341 score

H. Model Evaluation

The file "evaluate_model.py" evaluated the input model with DOPE potential. This profile was written to a file "TvLDH.profile", which was used as input to plot_profiles.py script to plot profiles with the Python matplotlib package. The GA341 score confirms that TvLDH.B99990005.pdb is a reasonable model.

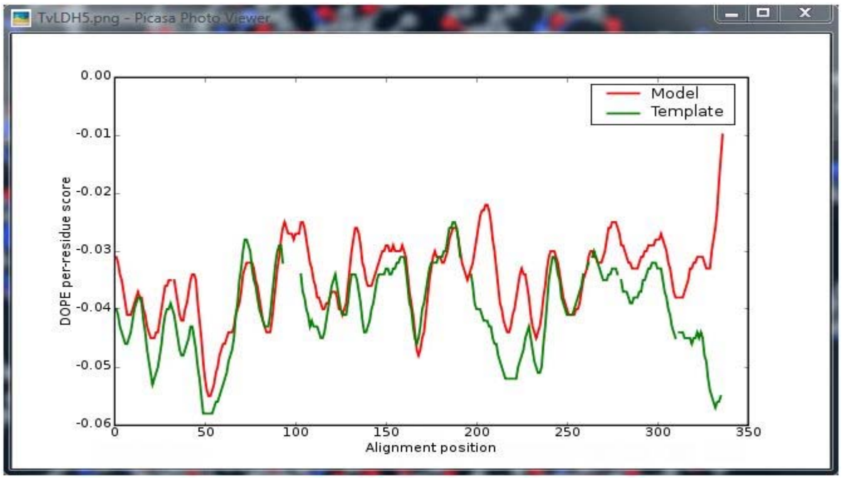


Fig.:- DOPE score plot of model.

I. Visualization of Model

The model so formed was visualized using JMol (an open-source Java viewer for chemical structures in 3D).

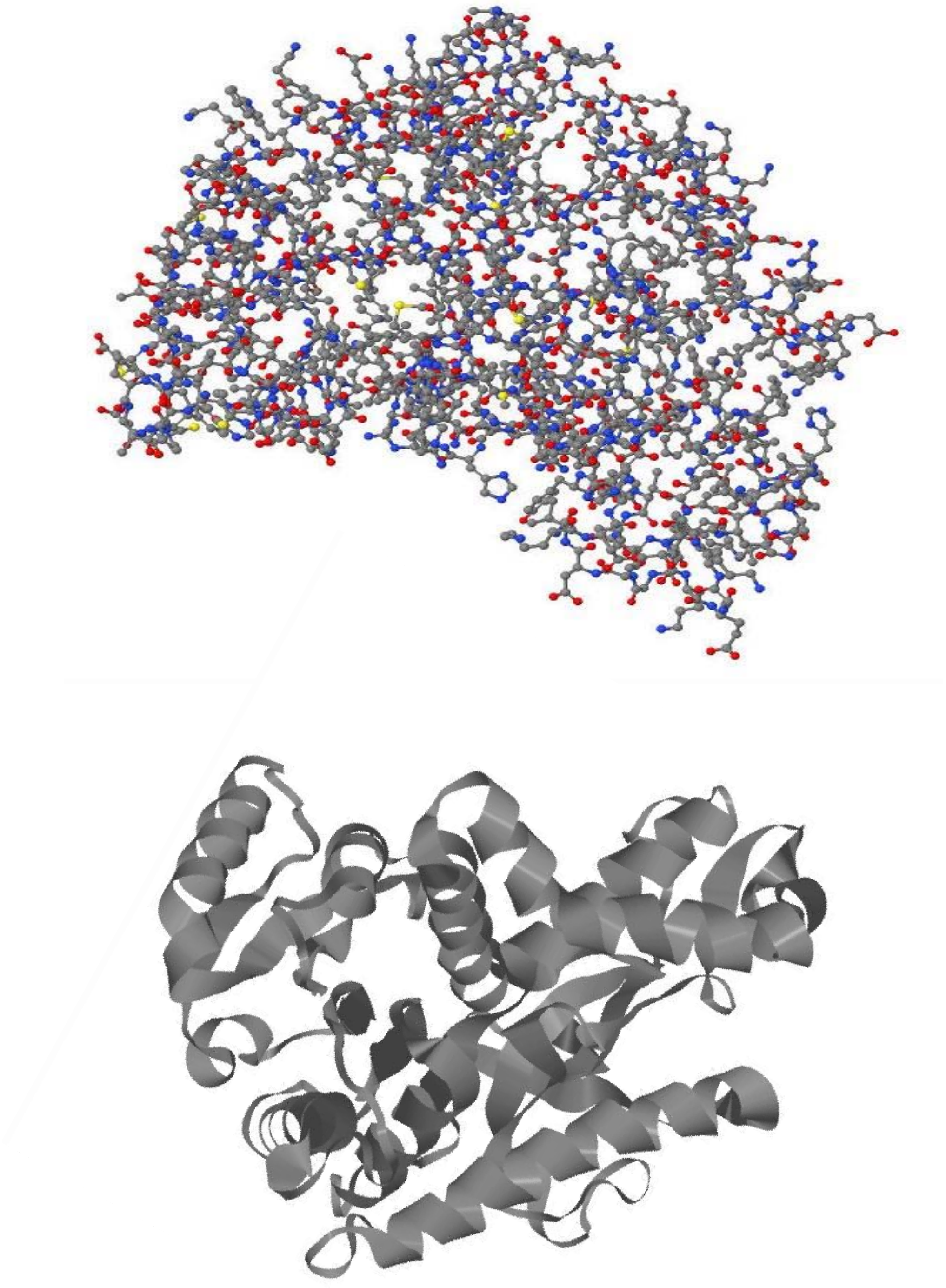


Fig.:- 3D structure of model in different views.

Time Saving Benefits

S.No.	Activity	Time taken by the MODELLER procedure	Time taken by the GUI procedure
1.	Save target sequence in .ali format	1 min	⚙
2.	Identify homologous structures using BLAST search	2 min 30 sec	⇔
3.	Selection and downloading of suitable Template(s)	5 min	⇔
4.	Opening compare.py and editing the template codes and chains	3 min	⚙
5.	Executing compare.py script using MODELLER	10 sec	⇔
6.	Opening compare.log, finding the required data and analyzing the dendrogram	2 min	⚙
7.	Selecting the template and editing codes in align2d.py	3 min	⚙
8.	Executing the align2d.py script using MODELLER	1 min 20 sec	⇔
9.	Opening the .pap file and analyzing the target-template alignment	2 min	⚙
10.	Opening and editing model-single.py for no. of models to be made and the alignment file	3 min	⚙
11.	Executing the model-single.py using MODELLER	8 min 35 sec (for 5 models)	⇔
12.	Opening model-single.log and analyzing the assessment score table at the end of the file for DOPE and GA341 score	2 min	⚙

13.	Selecting the model from the assessment scores, opening and editing evaluate_model.py script	3 min	⚙
13.	Executing the evaluate_model.py script using MODELLER	20 sec	↔
14.	Opening and editing evaluate_template.py for alignment file	3 min	⚙
15.	Executing the evaluate_template.py script using MODELLER	15 sec	↔
16.	Opening and editing plot_profiles.py for alignment file	3 min	⚙
17.	Executing the plot_profiles.py script using MODELLER	5 sec	↔
18.	Viewing the evaluation graph kept in the working directory	2 min	⚙
19.	Opening the model.pdb file with a visualising software like JMol	1 min	⚙
	Total Time Taken	46 min 15 sec	18 min 15 sec

⚙ Automated Process, takes less than a second. Thus, the time taken can be rendered negligible.

↔ The Activity takes the same time in the GUI process as in the MODELLER procedure due to original software dependency.

NOTE: All the time considered varies with user’s capability (of typing, analyzing, etc) and system’s configuration (RAM, CPU speed, etc).

CONCLUSION

- The application developed in this work can be used for easy homology modeling without knowing much about the complex MODELLER procedure and can proceed without any knowledge of scripting.
- The user does not have to worry about the input sequence formats and the alignment format that has to be supplied which is otherwise a very big problem while running MODELLER.
- Just pasting the sequence in the text window is a prerequisite; the rest of the process is taken care of by the application.
- Every step is automated, interactively guided and gives complete information of the steps of Homology Modeling.
- The models can be easily evaluated and their energy can be viewed by automated plotting feature.
- Thus the application provides a one place solution to all the homology modeling needs.
- The application does not require any kind of data to be shared anywhere.
- As demonstrated by the Time Saving Benefits table above, the developed application takes around 18+ minutes in comparison to the 45+ minutes taken by the process followed by original software to complete a task, saving around 25 – 30 minutes.

FUTURE SCOPE

Due to time constraint the following features could not be implemented in the GUI. We intend to do so as post-project work.

- Making the software more robust by proper error handling and displaying of messages.
- Allow the user to modify the Target-Template alignment.
- Feature of Loop refining of the built models. Amino acid residues may be mentioned where the DOPE energy profile shows differences between the template and the target sequence.
- Regular updating of the PIR database.
- Iterative modeling using temperature optimization.
- Releasing the application for platforms other than Windows.

TECHNICAL REQUIREMENTS

Hardware Requirements:-

- CPU :- Pentium 4 or higher.
- RAM:- 256 MB or more.
- 100 MB of Hard disk space

Software Requirements:-

- Operating System:- Windows (XP/ VISTA/ 7)
- MODELLER 9v7
- Python 2.3.5
- Java Runtime Environment
- Matplotlib 0.90.1
- Numpy 1.0.4
- JMol 11.8.9

REFERENCES

- **Roberto Sanchez and Andrej Sali**, (2000), Comparative Protein Structure Modeling, Methods in Molecular Biology, Humana Press Inc., 143:97-129
- **Mark A.Marti-Renom**, Comparative Protein Structure Prediction- MODELLER Tutorial, Prince Felipe Research Center (CIPF), Valencia,Spain
- **Andrej Sali and Tom L.Blundell**, (2003), Comparative Protein Structure Modelling by Satisfaction of spatial restraints, *Journal of Molecular Biology*, 234: 779-815.
- **M.S. Madhusudhan, Benjamin M. Webb, Marc A. Marti-Renom, Narayanan Eswar and Andrej Sali,** (2009), Alignment of multiple protein structures based on sequence and structure features, Protein Engineering, Design & Selection , 1–6
- **S. Altschul, W. Gish, W. Miller, E. Myers, and D. Lipman**, (1990), Basic local alignment search tool, *Journal of Molecular Biology*, 215:403-410.
- **Narayanan Eswar, David Eramian, Ben Webb, Min-Yi Shen and Andrej Sali,** (2006), Protein Structure Modeling With MODELLER, Current Protocols in Bioinformatics, John Wiley & Sons, Inc., 5.6.1-5.6.30
- **Schildt H.,** (2002), JAVA 2 : The Complete Reference , 5th Edition, McGraw Hill/Osbourne, pp. 1186
- **Horton I.,** (2005), Wrox – Beginning Java 2 JDK5, 5th Edition, Wiley Publishing, Inc., pp. 1501