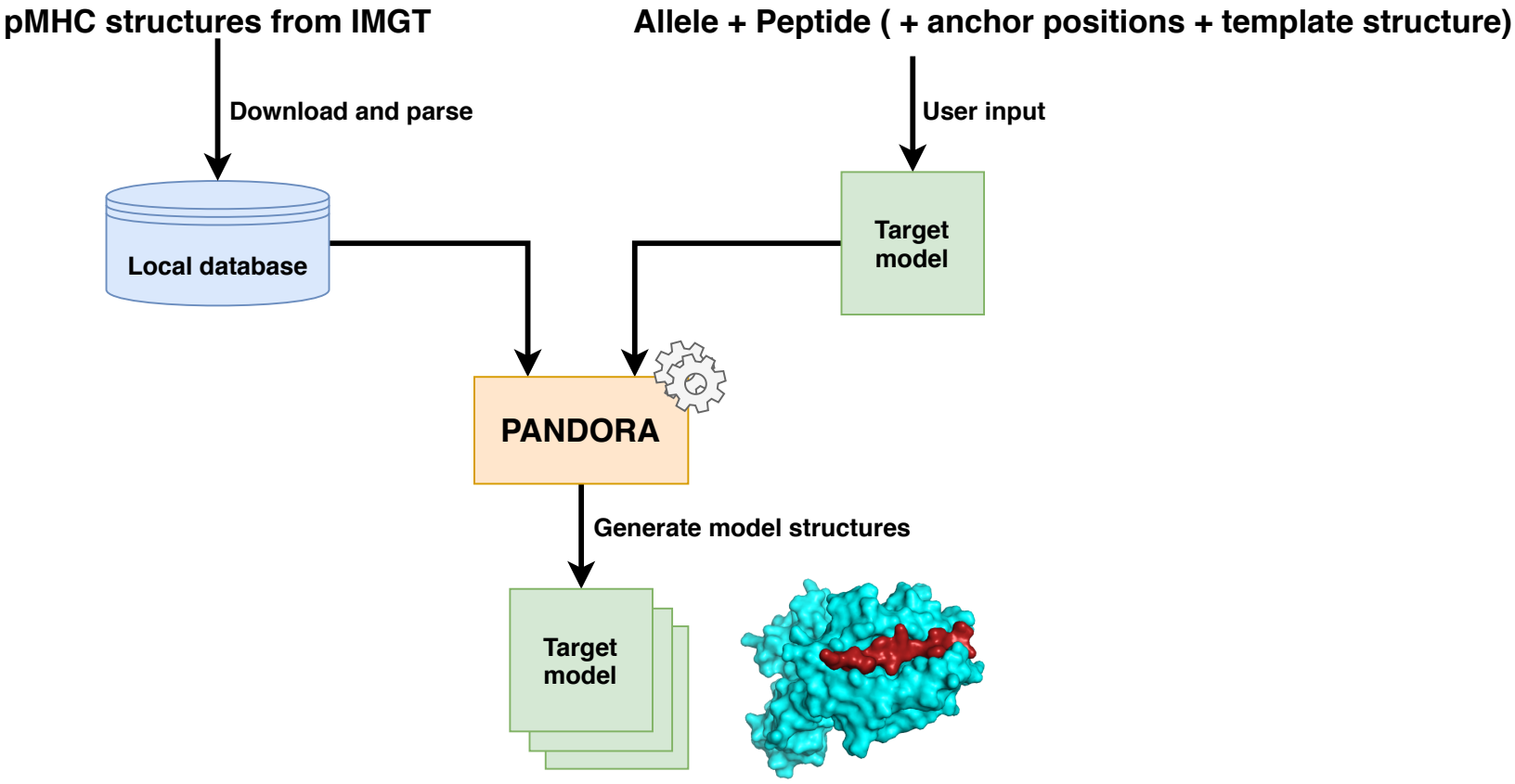
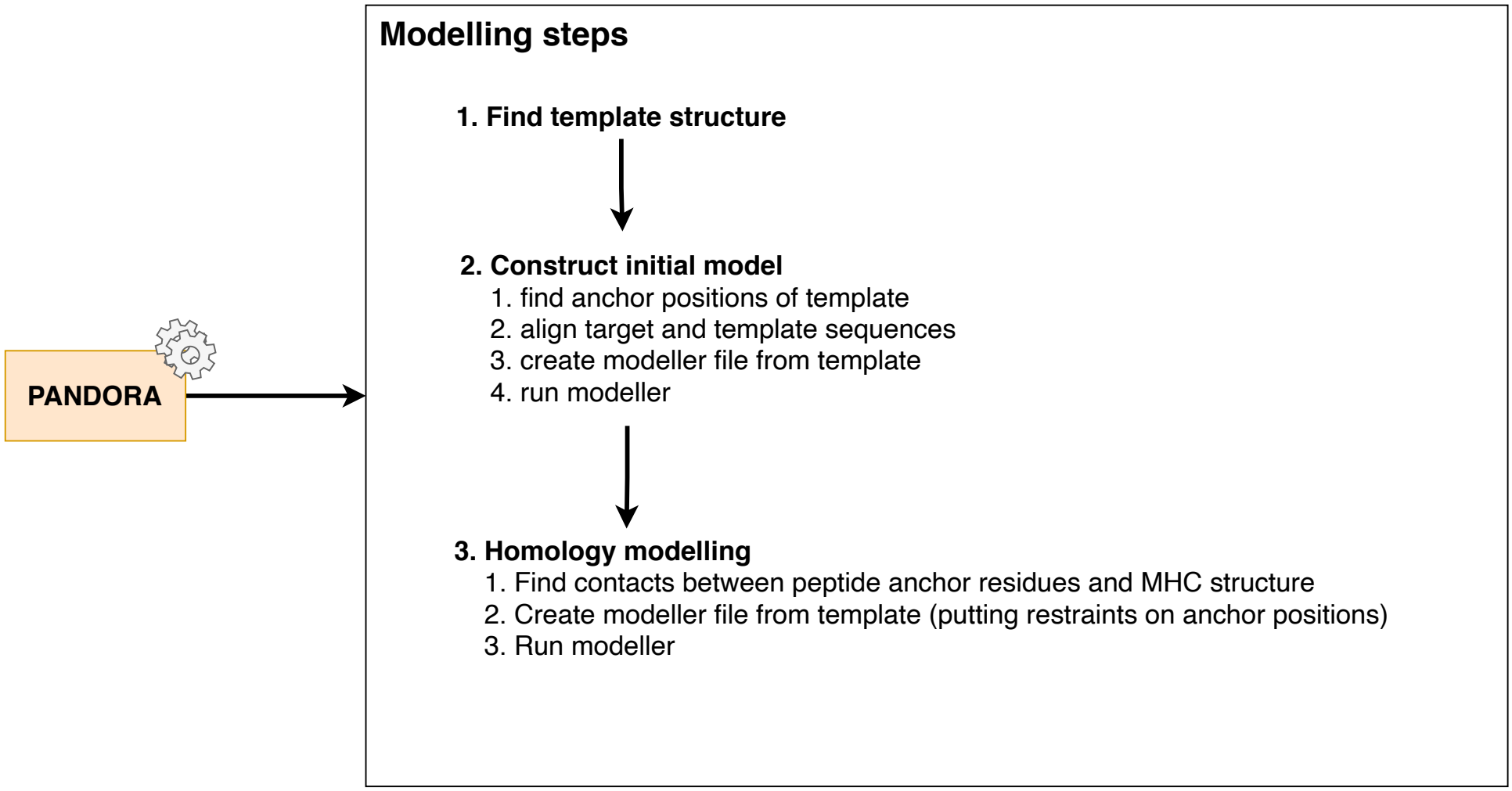
Flow

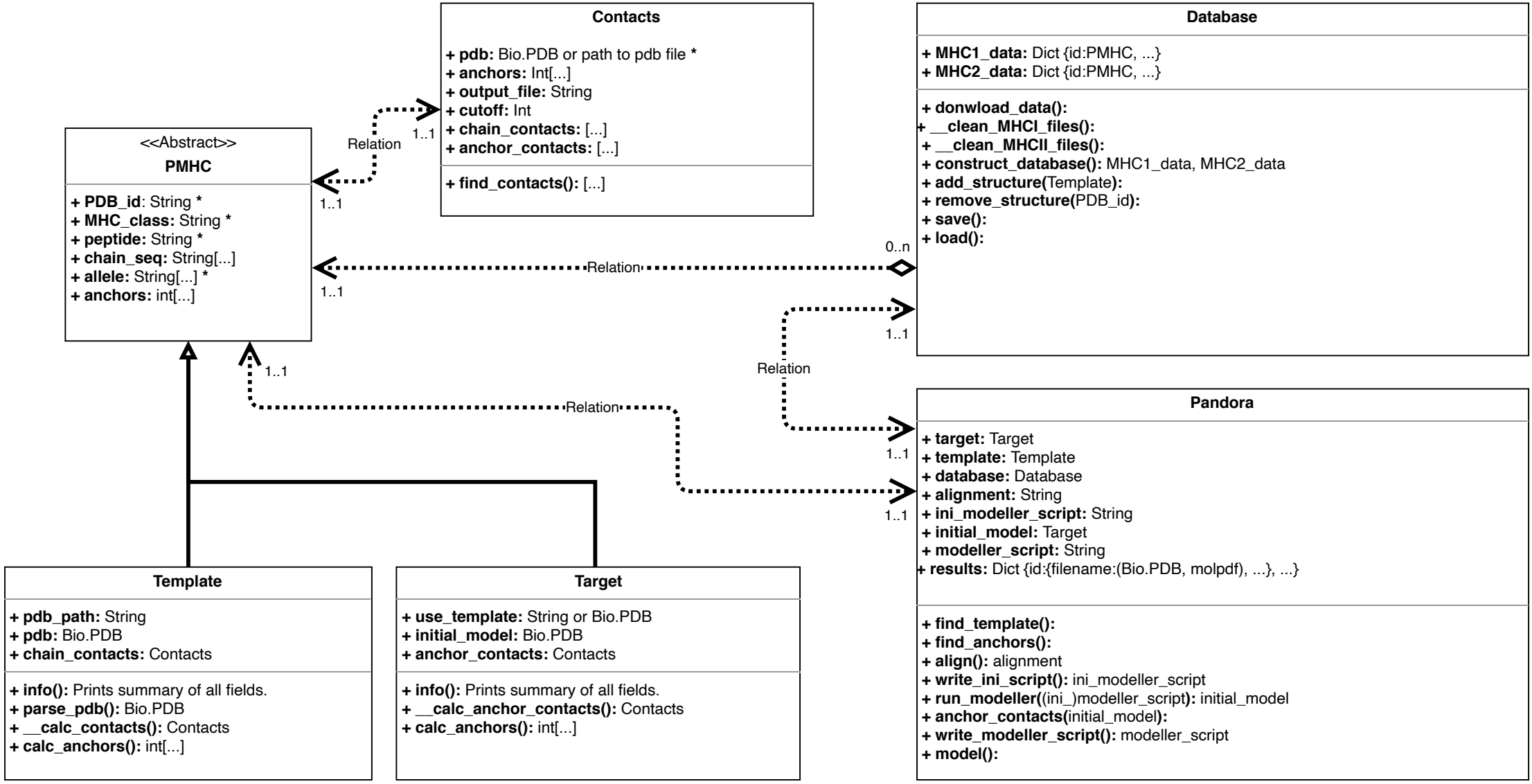
1. Build local database
2. Create target structure (user inputs allele and peptide + optional anchor positions)
3. Run Pandora




Modelling steps
1. Find template structure
2. Construct initial model
1. find anchor positions of template
2. align target and template sequences
3. create modeller file from template
4. run modeller
3. Homology modelling
1. Find contacts between peptide anchor residues and MHC structure
2. Create modeller file from template (putting restraints on anchor positions)
3. Run modeller
PANDORA

Classes

* = required input field



Explanation of the methods

PMHC

This class stores the information of a template or target model.

info(): This method prints a summary of all fields
calc_anchors(): The user can define the anchors, but this function can also calculate them based on a pdb file.

----- Template -----

parse_pdb(): This function takes the path to a pdb file as input, loads this pdb file as a Bio.PDB object. If the user did not specify alleles, or chain/peptide sequences, they are inferred from the pdb file.
__calc_contacts(): calls **Contacts.find_contacts(**pdb**)** to find the chain contacts.

----- Target -----

__calc_anchor_contacts(): calls **Contacts.find_contacts(**pdb, anchors = [...]**)** to find the peptide anchor - MHC contacts.

Contacts

This class calculates atom contacts.

find_contacts(): calculates all atom contacts in a pdb file. The results are stored in a list or can be written to a file. If the user provides anchor locations (as a list), the contacts between the peptide anchor residues (based on the given anchor locations) and the rest of the MHC structure are calculated. This is used for putting restraints in modeller.

Database

This class is builds the database that contains all template structures. It performs downloading and cleaning/parsing. It also functions as a neat way to store all template for later use.

download_data(): Downloads all data from IMGT
__clean_MHCI/II_files(): Cleans the downloaded files
construct_database(): Create a dict containing all templates
add_structure(): Add a single structure to the database. This allows the user to manually add extra structures to the db.
remove_structure(): Remove a structure from the database
save():, load(): Save or load a pickle file of the database for later use.

Pandora

This class performs all modelling, either in separately steps or in a single wrapper function.

find_template(): Finds the best template structure in the local_database suitable for modelling the target. This is based on the target allele and peptide. The user can also give a specific template structure if he/she wants to use their own structure as template.
find_anchors(): Calculates the anchor positions of the template structure.
align(): Performs an alignment of the target and template structure, this is needed for modeller to create the initial model.
write_ini_script(): Write a modeller python script to perform the modelling of the initial model that is refined in later steps.
run_modeller(): Run modeller based on a modeller script. This is either the script from **write_ini_script()** or **write_modeller_script()**. From the initial modelling run, a .ini file is created, which is a .pdb file in reality. The **initial_model** field of the Target is updated with this file.
anchor_contacts(): Calculate the contacts between the anchors and the structure. This just calls the **__calc_anchor_contacts()** from the Target object.
write_modeller_script(): Writes the modeller python script for the real homology modelling. Contacts of the anchor positions in the Target object are used to put restraints on the peptide. After running this method, **run_modeller()** is called to do the modelling.
model(): Wrapper function that performs all previously mentioned modelling functions in order.

File Structure

```
Pandora_0.0.1
    >PANDORA
        >run
            -benchmark_model.py              #(script for benchmarking by looping through all templates)
            -neoantigen_model.py             #(script for modelling new pMHC models)
        >core
            -core.py                         #core classes of PANDORA
        >modelling
            -modelling.py                    #(functions for modelling)
            -cmd_modeller_ini.py             #(template script for modelling)
            -cmd_modeller.py                 #(template script for modelling)
            -MyLoop_template.py              #(template script for modelling)
        >parsing
            -utils.py                        #(misc. functions)
            -url_protocols.py                #(functions to download data)
            -structure_parser.py             #(functions to clean/parse downloaded data)
            -get_pMHC_anchors.py             #(functions to find anchor positions)
        >tools
            -pdb_atom_contacts.py.           #(class for calculating atom contacts)
        >PANDORA_files
            >database                        #(database of template structures is stored here)
            >outputs                         #(output/temp files of the target model)
            >PDBs
                >IMGT_retrieved              #(contains the downloaded .gz data from IMGT)
                >pMHCI                       #(contains the cleaned pdb files for MHCI)
                >pMHCII                      #(contains the cleaned pdb files for MHCII)
         -__init__.py

    >test
        -test_functions.py                   #(functions to test all functions)
        -run_test.py                         #(run all test using test_functions.py)

    >example
        -pandora_example.py                  #(example script, tests full functionallity of the pipeline)
     -setup.py
     -LICENCE
     -README
```

Example Python code

```
import PANDORA

## 1. Create local database
db = PANDORA.Database.construct_database()

## Alternatively, load pre-built database
db = PANDORA.Database.load('db.pkl')

## 2. Create target object
target = PANDORA.PMHC.target(PDB_id = '1A6A', MHC_type = 2,
                             allele = ['HLA-DRA*0102','HLA-DRB1*0301'], peptide = 'PVSKMRMATPLLMQA')

## 3. Perform modelling
model = PANDORA.Pandora.model(target, db)
```