Midterm Coursework: RESTful Web Service

Introduction

During the course so far we have developed a number of database backed web servers using Django. For this assignment you are tasked with developing a RESTful web service using the knowledge you have gained so far.
The server should use Django Models to appropriately design and model the input data and provide useable migrations for the underlying database. As per the examples in the lesson you will be working with a small dataset of biological data, the CSV files of the data and a document describing the data is provided so you can design an appropriate model.
To successfully implement the application you should have a good understanding of the flow of data through a Django application. You should make appropriate use of Serializers, forms and validation. You should ensure you make appropriate use of Django Views. The application should be laid out in a manner that is clear and easy to follow.
This assignment is worth 50% of the total mark for this module

Task

You are working with a group of BioScience researchers who are working on protein domains. They need a system that they can regularly query for the data they've generated. It has been decided that a RESTful web service will be implemented and you have been asked to build that service.
The researchers have given you 3 files. The first is a file of protein amino acid sequences. Each protein has and ID and an animo acid sequence.
The animo acid sequence is represented using a convenient 20 letter code. The second file is a series of domain annotations the researchers have created. The final file is data about the domains. They would like you to build a RESTful web application that will store this data and which they can query to get this data out of it. The application should store the data, in the 3 provided files, in a database and return json documents of the following the specification outlined in the REST specification document.

Deliverables

1. A zip file of the Django application that implements the REST API as described in the accompanying REST specification

2. The instance of the Django application in the zip file should have all the data loaded

3. The zip file should contain a python script or method for loading the provided CSV data in to the database

4. A brief report explaining the code in your application and how it meets the requirements. You may use focused, short code extracts if they make your explanation clearer. Explain the logic of your approach, why is your code arranged as it is? This report should also include how to run the unit tests and the location of the data loading script.

Requirements

We will assess your work based on the following requirements and criteria:
R1: The application contains the basic functionality describe in class

a) correct use of models and migrations

b) correct use of form, validators and serialisation

c) correct use of django-rest-framework

d) correct use of URL routing

e) appropriate use of unit testing

R2: Implements an appropriate data model for the data

R3: Implementation of appropriate code for the required REST endpoints
R4: Implementation of appropriate method of bulk loading data

Code style and technique

Your code should be written according to the following style and technique guidelines:

C1: Code is clearly organised into appropriate files (i.e. view
code is placed in an appropriate view.py or api.py file, models are placed in
an appropriate models.py file)

C2: Appropriate comments are included to ensure the code is clear and readable

C3: Code is laid out clearly with consistent indenting, ideally following python pep8 standard
C4: Code is organised into appropriate functions with clear, limited purpose

C5: Functions, classes and variables have meaningful names, with a consistent naming style
C6: Appropriate Unit tests to cover the REST API functionality are provided
Submission

You should write a brief report and submit your source code. The submission should contain the following items and information:
D1: Django code in standard ZIP format

D2: Short report in PDF format. Including how to unpackage and run your application and how to run the tests for your application. Explain where it can be found in the code.

Marking Criteria

We will mark your work according to the set of criteria shown below,
which consider the requirements, your programming technique and style and the documentation you have provided:

Code Style & Technique

C1: Code is clearly organised into appropriate files (i.e. view code is placed in an appropriate view.py or api.py file, models are placed in an appropriate models.py file)
C2: Appropriate comments are included to ensure the code is clear and readable
C3: Code is laid out clearly with consistent Code Style & Techniqueindenting, ideally following python pep8 standard
C4: Code is organised into appropriate functions with clear, limited purpose
C5: Functions, classes and variables have meaningful names, with a consistent naming style

C6: Appropriate tests to cover the API functionality are provided

Requirements

R1: The application contains the basic functionality described in class
R2: Implements and appropriate data model for the data
R3: Implementation of appropriate code for the required REST endpoints
R4: Implementation of appropriate method of bulk loading data

Submission

You should use Djangos default SQLite3 database!

D1: Django code in standard ZIP format
D2: Short report in PDF format. Including how to unpackage and run your application and how to run the tests for your application. For each requirement (R1 → R4) state how this was achieved or if it was not achieved. Explain where it can be found in the code. Use focused, short code extracts if they make your explanation clearer.