Texas senate database

Team: For-git about it

Contents

[Introduction 2](#__RefHeading__260_1789256170)

[Problem 2](#__RefHeading__262_1789256170)

[Use Cases 2](#__RefHeading__264_1789256170)

[Django Models 3](#__RefHeading__266_1789256170)

[Senator Model 3](#__RefHeading__268_1789256170)

[Bill Model 3](#__RefHeading__270_1789256170)

[Committee Model 3](#__RefHeading__272_1789256170)

[Vote Model 3](#__RefHeading__274_1789256170)

[Senator and Bill Relationship 3](#__RefHeading__276_1789256170)

[Senator and Committee Relationship 3](#__RefHeading__278_1789256170)

[Bill and Committee Relationship 3](#__RefHeading__280_1789256170)

[API 3](#__RefHeading__282_1789256170)

[UML Diagrams 3](#__RefHeading__284_1789256170)

# Introduction

## Problem

The democratic process works best if there is a well-informed citizenry. Given more information, voters can make more informed decisions on their support for political candidates. However, there currently does not exist a well-made central database containing information concerning the Texas Senate. There is no place to see comprehensive information on bills, senators, committees and the relationship between them. Without such a database, it is difficult for citizens to hold their government accountable. The ability for voters to see a candidate’s voting record allows them to decide for themselves whether a candidate is suitable for the job. While such databases exist for the federal government (<https://www.govtrack.us/> is an example), the only place where one can access a database for bills going through the Texas legislature is through the official Texas legislature website, which is lacking in information. The Texas Senate Database exists to provide such a service, to open source democracy in Texas, and to serve as a source of information so that voters can learn about bills, legislators, and committees in order to stay informed about the democratic process in Texas.

## Use Cases

The Texas Senate Database contains the following things:

Information about specific bills that are going through, or have gone through, the Texas Senate, including a brief summary, author, status, a link to the full text of the bill, the committee that the bill went through, and a breakdown of the votes for the bill.

Information about individual senators, including a brief biography, their voting history, a list of committees they chair or are a member of, and a link to their facebook and/or twitter, if they have one.

Information about Senate committees, including a description, their members, and a list of all bills to have gone through that committee.

Users of the Texas Senate Database can learn about the most recent bills to go through committee or a floor vote, or they can look through the lists of senators and committees and see information about the bills they have voted on.

Users can use the Texas Senate Database to see Senator’s voting records over time, and thus get a better understanding of a senator’s position on issues. Voters can then use this information to affect their voting decision.

Users can see a list of bills that passed or failed and gain a greater understanding of the legal trends for the Senate as a whole. Users in the political field can use this information to make informed predictions on proposed bills or bills to be proposed.

The net effect of this information is to allow voters in Texas to contribute to and to learn about the Texas senate, to inject some much needed openness in the democratic process, and to give voters and citizens in Texas the tools to make informed decisions about their Senate.

# Django Models

## Senator Model

The senator model models each of the 31 individual senators in the Texas Senate. These are the legislators in the Texas Senate who author, sponsor, and vote on bills, whether in committees or on the floor. The primary key is an auto-generated integer ID supplied by Django. Attributes include their name, district, party, occupation, legislative experience, and their facebook and twitter pages if they have one.

## Bill Model

The bill model models specific bills submitted to committees or the senate floor at large. These bills are bills that have gone to a vote in committees or on the floor. As with senators, their primary key is an auto-generated integer supplied by Django. Attributes include the name of the bill, the legislative session in which the bill was proposed, the date it was proposed, and if passed, the date it was signed into law and the date it was effective. Attributes also include its status, a link to its full text, and a brief description on the contents of the bill.

## Committee Model

The committee model models the 31 standing committees in the Texas Senate. Every bill must go through committee, where it is debated, amended, and voted on before it goes to the floor where it is voted on by the Senate as a whole. As with senators and bills, the primary key is an auto-generated integer supplied by Django. Attributes include its name, a description, and the date that the committee was first formed.

## Vote Model

???

## Senator and Bill Relationship

Bills are authored by Senators, and although there are co-authors, we have chosen to only include the main author, for simplicity’s sake. A bill can have a long list of co-authors, some of which may not be senators, and thus, as an initial step, we have decided only to include main authors. Senators can author multiple bills. Therefore, we have a many to one relationship between bills and senators.

## Senator and Committee Relationship

There are 31 standing committees in the Texas Senate. Each committee is chaired by a senator and vice chaired by a senator, though those positions may be vacant at any given time. Each committee also has some number of senators as members, usually less than 10. A senator may be part of any number of committees. Therefore, there exists a many to many relationship between senators and committees.

## Bill and Committee Relationship

Each bill must go through a committee. While a bill may go through multiple committees before going to the floor for a vote, and indeed, every bill must go through the calendar committee to be placed on the floor for a vote, each bill mainly falls under the purview of one committee. This is the committee which we have chosen to model. Each committee oversees many bills. Therefore, there exists a many to one relationship between bills and committees.

# API

The API splits up data into three sections: senators, committees, and bills.

* /api/senators – GET – Gets a collection of all of the senators in the database
* /api/senators – POST – Adds a new senator to the collection of senators
* /api/senators/{id} – GET – Gets a specific senator, where {id} is the identification key of the desired senator
* /api/senators/{id} – PUT – Updates a specific senator, where {id} is the identification key of the desired senator
* /api/senators/{id} – DELETE – Delete a senator from the database, where {id} is the identification key of the senator to remove
* /api/senators/{id}/committees – GET – Gets all of the committees that the senator with id {id} is a member of
* /api/senators/{id}/bills – GET – Gets all of the bills that the senator with id {id} has authored
* /api/bills – GET – Gets a collection of all of the bills in the database
* /api/bills – POST – Adds a new bill to the collection of bills
* /api/bills/{id} – GET – Gets a specific bill, where {id} is the identification key of the desired bill
* /api/bills/{id} – PUT – Updates a specific bill, where {id} is the identification key of the desired bill
* /api/bills/{id} – DELETE – Delete a bill from the database, where {id} is the identification key of the bill to remove
* api/bills/{id}/senators – GET – Gets a list of senators and their corresponding votes for the bill with id {id}
* api/bills/{id}/authors – GET – Gets a list of the senators who authored the bill with id {id}
* /api/committees – GET – Gets a collection of all of the committees in the database
* /api/committees – POST – Adds a new committee to the collection of committees
* /api/committees/{id} – GET – Gets a specific committee, where {id} is the identification key of the desired committee
* /api/committees/{id} – PUT – Updates a specific committee, where {id} is the identification key of the desired committee
* /api/committees/{id} – DELETE – Delete a committee from the database, where {id} is the identification key of the committee to remove
* /api/committees/{id}/senators – GET – Get all the senators in the committee with id {id}
* /api/committees/{id}/bills – GET – Get all the bills originating in the committee with id {id}

## API Unit Tests

Unit tests for the API use the Python unittest library to individually test each endpoint and the associated HTTP methods. At the start of each test method we open a connection to the Apiary-mock servers which we close at the end of the method. In each test, the response status of the HTTP request is checked. In the methods that return a response body, the body is in byte form and must be decoded to a string and then loaded into JSON using the “json.loads” function.

In the tests for the POST and PUT HTTP methods, the variable “values” is used with the “json.dumps” function to convert a python list/dictionary into a JSON object to be passed to the client. For the POST method, the response body is a JSON object containing only the id of the new object. For the PUT method, there is no response body check.

In the tests for the GET HTTP methods, the variable “desired\_body” contains the expected response body in the form of a python object that is checked against the response returned from the client.

In the tests for the DELETE HTTP methods, there is no “desired\_body” or “values” variables as no JSON objects are passed to or from the client.

# UML Diagrams

Info goes here