Quick Deploy Database API

By

Daniel Hickenlooper and Paul Kreiner

Submitted to
The Faculty of the School of Information Technology
in Partial Fulfillment of the Requirements for
the Degree of Bachelor of Science
in Information Technology

Daniel Hickenlooper	4/17/17 Date
Paul Kreiner	4/17/17 Date
Brian Verkamp, Faculty Advisor	4/17/17 Date

University of Cincinnati
College of Education, Criminal Justice, and Human Services

April 2017

CONTENTS

TABLES AND FIGURES

ABSTRACT

The use of an Application Program Interface (API) allows organizations to open their information for public consumers. This allows developers a shortcut to creating applications with a previously built data set. However, for individual developers or small businesses, the process of creating an API for data access is overlooked as a costly and time-consuming process. With our repository of pre-configured, open-source packages, these users can now set up an API for their local database with minimum additional configuration. Once deployed, all current and future applications, widgets, and websites will have access to the database through the API over HTTPS without additional database management.

1. PROBLEM STATEMENT

1.1 Introduction

This project is being completed for the University of Cincinnati School of Information Technology. All products used in this project are either supplied by the University of Cincinnati for educational purposes, or provided for free online as open-source products.

1.2 Project Description

We created an open-source Application Program Interface (API) for use on internal databases. This API is presented as a bundle of open-source technologies, meaning they are created by the open-source community and available for no cost. This bundle is created using a GitHub repository, which contains the MIT open-source license and is available for public cloning. Via a Representational State Transfer (REST) environment, objects on the database will be accessible through Hypertext Transfer Protocol (HTTP). For users who do not have a database ready, a SQLite database will be available to build with the API.

1.3 Problem

Smaller businesses and independent developers currently lack the resources to create an API, or spend too much time connecting multiple applications to a database without the use of an API.

1.4 User Profile

1.4.1 Potential Users

Potential users for this project include small businesses or start-ups, independent developers, and medium-large businesses. However, due to the ease and cost of the product, the target user base will be small businesses or start-ups and independent developers.

a. Small Businesses and Start-ups

In order to quickly grow their business while keeping costs to a minimum, small businesses and start-ups will be able to use our pre-configured package of open source tools to create an API for their database. This will allow them to quickly create and deploy applications with their current data. Similar products include cloud technologies and a fee, which these businesses may not be able to afford in their current capacity. With our product, we can provide a friendly stepping stone with little configuration on the user's end.

b. Independent Developers

Independent developers are more likely to create applications or websites for their own personal use than for profit. Developers may find the setup of an API for their data a tedious process before being able to develop their applications. With our package, the time to set up the back-end will decrease and developers can begin using their programs quicker. Should they want to alter the API in any way, the open source framework allows changes to be made without fear of breaching license agreements.

1.4.3 Software and Interface Experience

Users will need a basic understanding of command line operations for their chose operating system and communication over HTTP. While documentation will attempt to make this as simple as possible, the experience will help accelerate deployment.

1.4.4 Experience with Similar Applications

N/A – The goal is that the user will not need to have previous API experience in order to deploy the package.

1.4.5 Task Experience

HTTP communication between apps, widgets, and websites with the backend database through the REST API.

1.4.6 Frequency of Use

The package will be deployed once per database. Once deployed, all apps, widgets, and websites will be able to communicate with the database through the newly established API.

1.4.7 Key Interface Design Requirements

All interfaces used in this product are standard on Window's machines. Users will use the command line to install, any web browser to test API connectivity, and any text editor to change the table schema within the Python application file.

2. PROJECT MANAGEMENT

2.1 Objectives/Deliverables

Infrastructure Build – 10/10/2016

Open-Source API Build – 11/21/2016

API Deployment Repository – 4/18/2017

API Build Documentation - 4/18/2017

2.2 Budget

The budget is based on retail prices and an estimated value for labor. Since this project is being done for education purposes and using the School of Information Technology Sandbox, the actual total cost is zero dollars. The estimate is a potential cost for business use.

No	Item	Unit	Price	Total
Hardware				
1	Server	1	$1,000.00	$1,000.00
2	Labor	50	$70.00	$3,500.00
Software				
2	Labor	150	$70.00	$10,500.00
	Total			$15,000.00

Table 1: Estimated Budget

2.3 Project Schedule

2.3.1 Timeline

Below is an estimated timeline of completion for each step.

Task Name	Duration	Start	Finish
Secure Virutal Environment	3 days	Mon 9/12/16	Wed 9/14/16
Define Need/Client	8 days	Thu 9/15/16	Mon 9/26/16
Build PROD Infrastructure	7 days	Mon 9/19/16	Tue 9/27/16
Document Infrastructure Requirements	76 days	Mon 9/19/16	Sat 12/31/16
Build DEV Infrastructure	7 days	Mon 9/26/16	Tue 10/4/16
Choose and Implement REST Environment	7 days	Mon 10/3/16	Tue 10/11/16
Choose and Implement Python IDE	7 days	Mon 10/3/16	Tue 10/11/16
Choose and Import Public Database	7 days	Mon 10/10/16	Tue 10/18/16
QA Environment Deployment and Testing	21 days	Tue 10/18/16	Tue 11/15/16
PROD Environment Deployment	7 days	Tue 11/15/16	Wed 11/23/16
QA Integrate Database and REST Environment	14 days	Tue 11/15/16	Fri 12/2/16
PROD Integrate Database and REST Environment	7 days	Sat 12/3/16	Mon 12/12/16
Build Python API	77 days	Tue 12/13/16	Wed 3/29/17
Document API Build	77 days	Tue 12/13/16	Wed 3/29/17
Initial Testing	7 days	Sun 1/15/17	Mon 1/23/17
Document API Deployment and Useage	37 days	Wed 2/1/17	Thu 3/23/17
BETA Testing	7 days	Sun 3/12/17	Mon 3/20/17
Upload to Git Repository	7 days	Thu 3/23/17	Fri 3/31/17
Git Deployment Testing	7 days	Thu 3/23/17	Fri 3/31/17
Final Configuration/Updates	7 days	Sun 3/26/17	Mon 4/3/17
Prepare for Presentation	14 days	Sun 3/26/17	Wed 4/12/17

Table 2: Timeline

2.3.2 Gantt Chart

The Gantt Chart reflects the timeline in a more visual form to show what will be worked on in parallel.

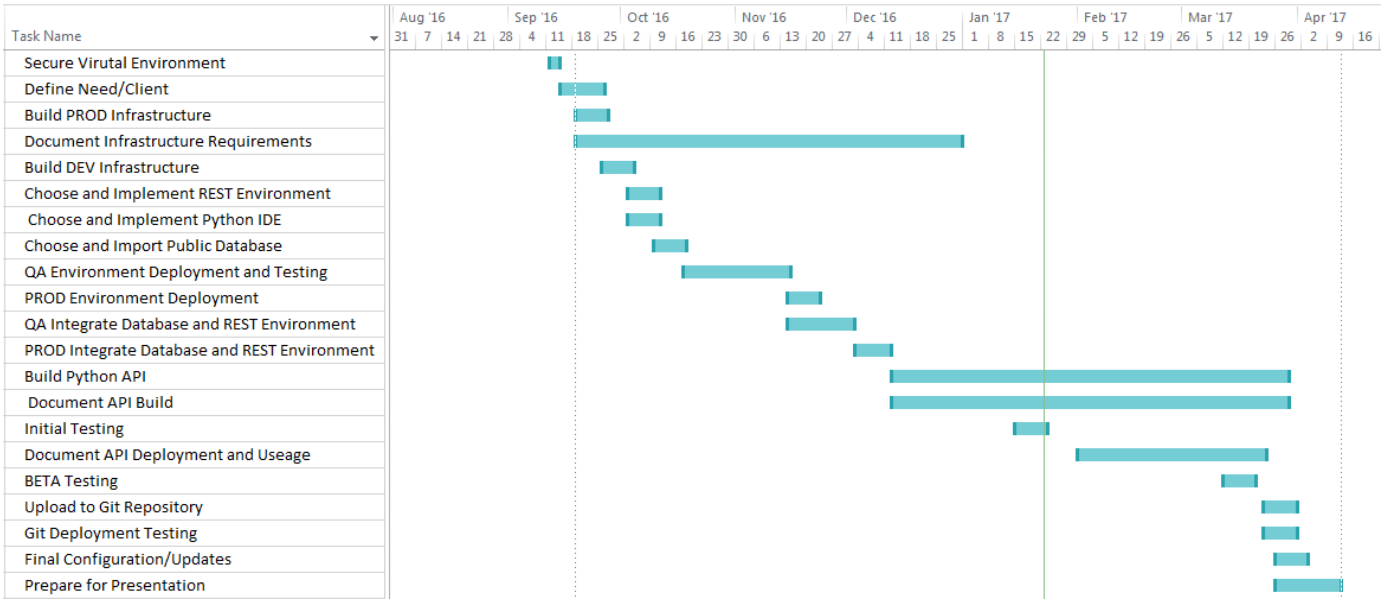

Figure 1: Gantt Chart

2.4 Use-Case Diagram

The diagram below shows the typical user interaction with the API.

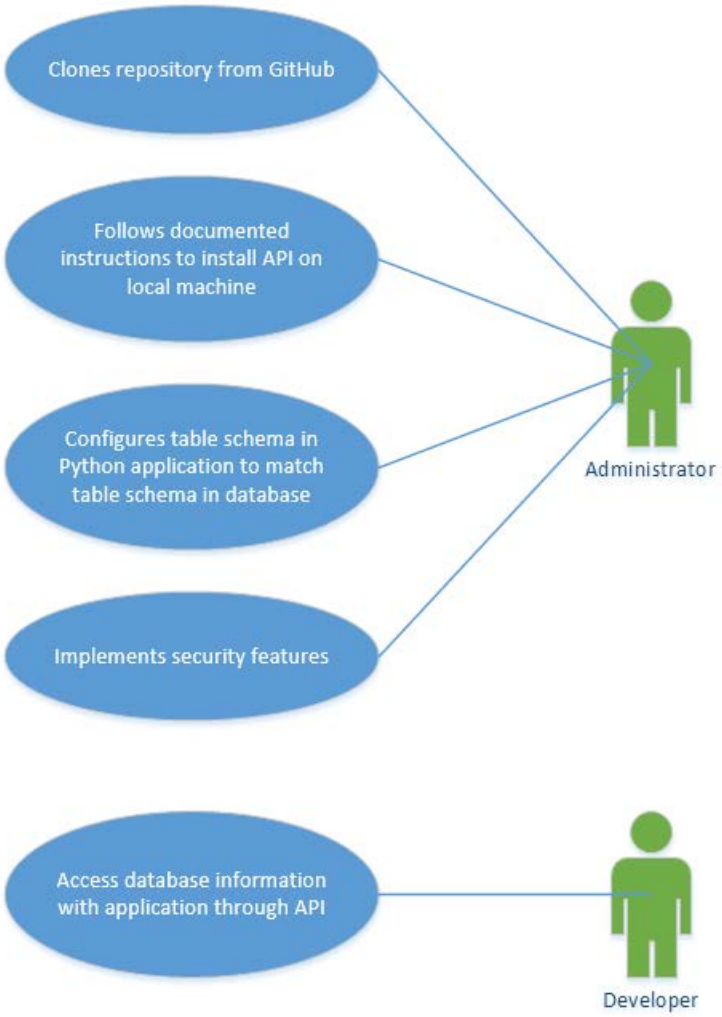

Figure 2: Use-Case Diagram

3. TECHNICAL ELEMENTS

3.1 Network

A Local Area Network (LAN) connection is required to connect the database and the API, and subsequently to access the API. An internet connection is required in order to download the open source materials, and initially to clone the repository from GitHub.

3.2 Sandbox

A sandbox environment will be used for testing purposes to maintain a low budget. The setup will consist of two virtual machines: a Windows machine to host the database and the API, and a Windows client to access the database through the API.

3.3 Software

The operating system used for testing is Windows 10 for both the client and server machines. Otherwise, only open-source software will be used for this project. The list below will be updated as new software is introduced to the project.

Software List:

- Python - https://www.python.org/
- Flask - http://flask.pocoo.org/
- Windows 10
- MySQL - https://www.mysql.com/
- SQLite - https://www.sqlite.org/
- VirtualEnv - https://virtualenv.pypa.io/en/stable/

3.4 Infrastructure Diagram

The diagram below shows our base setup for our environment. Users can choose between the two setups depending on whether they have a database available or not.

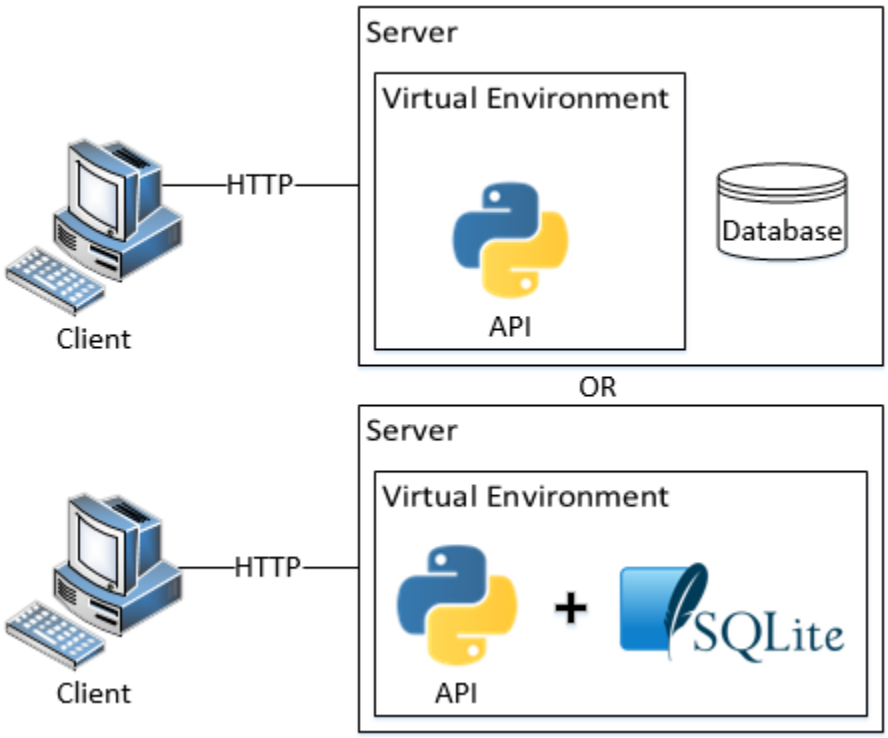

Figure 3: Infrastructure Diagram

4. TESTING

4.1 Overview

This section is to be used by team members, developers, and testers to make sure the testing of the project goes correctly and bugs are able to be documented and fixed. The testing will be done up to the Tech Expo so all of the problems are found and corrected, and the project works as intended.

4.2 Test Plan

The main testing for this project is on the Python API to ensure the program and the connections it requires are working correctly. There will also be tests run on the database to be sure the entries are correct and being kept in the correct fields.

These tests will be run after each project update to be able to target bugs and ensure the program is working before further progressing.

4.3 Test examples

Python

- Code runs
- API connection to web browser
- Reading from database
- Writing to database

Database

- String field accepts correct input
- Numeral field accepts correct input

4.4 Bug reporting

If a bug is found during testing, the member will log the bug and the effect the bug had on the program. The team will then work to fix the bug, and test the program again to ensure the bug is fixed, no new bugs are found, and it is working correctly.

5. CONCLUSION

The quick-deploy database API package is a great tool for developers with relatively small amounts of data, or no data, that want to skip over performing database administration items. The collection of open-source tools allows for a very low-cost option in connecting multiple applications to a single source of information, and the MIT license attached allows customization for different projects. This package of material is a good base for individuals who are exploring the use of APIs for their applications, but still allows them to add additional features as they grow in knowledge to fit their needs.

The repository can be found at: http://www.github.com/kreinepa

6. BIBLIOGRAPHY

Ronacher, Armin and Contributors. "Flask (A Python Microframework)." *Welcome | Flask (A Python Microframework)*. Armin Ronacher, 2015. Web. 28 Nov. 2016.

Bayer, Michael and Contributors. "The Python SQL Toolkit and Object Relational Mapper." *SQLAlchemy*. MIT, 15 Nov. 2016. Web. 28 Nov. 2016.

7. SOFTWARE SOURCES

Python - https://www.python.org/

Flask - http://flask.pocoo.org/

MySQL - https://www.mysql.com/

SQLite - https://www.sqlite.org/

VirtualEnv - https://virtualenv.pypa.io/en/stable/

8. SPECIAL THANKS

Special thanks to our advisors James Scott, Brian Verkamp, and Patrick Kumpf. To Nicholas Tate of GE Corporate for facilitating the idea for the project. To Kevin Ernst of the University of Cincinnati College of Engineering & Applied Science for facilitating with the Python code work.