INSTALLING PYTHON

CHECK IF YOU ALREADY HAVE PYTHON 3 INSTALLED

- Inside your “terminal” for Mac, or “command prompt” for windows.
  - Type “python –-version”

PYTHON INTEGRATED DEVELOPMENT ENVIRONMENT

WHAT ARE IDES?

- IDEs are simply a source code editor. This means an IDE will help and assist in writing software!
- Many of them have terminals and other useful build automation tools embedded within.
- IDEs have debugger tools that allow you to dive deep within the code to find bugs and create solutions

WHY DO DEVELOPERS USE IDES?

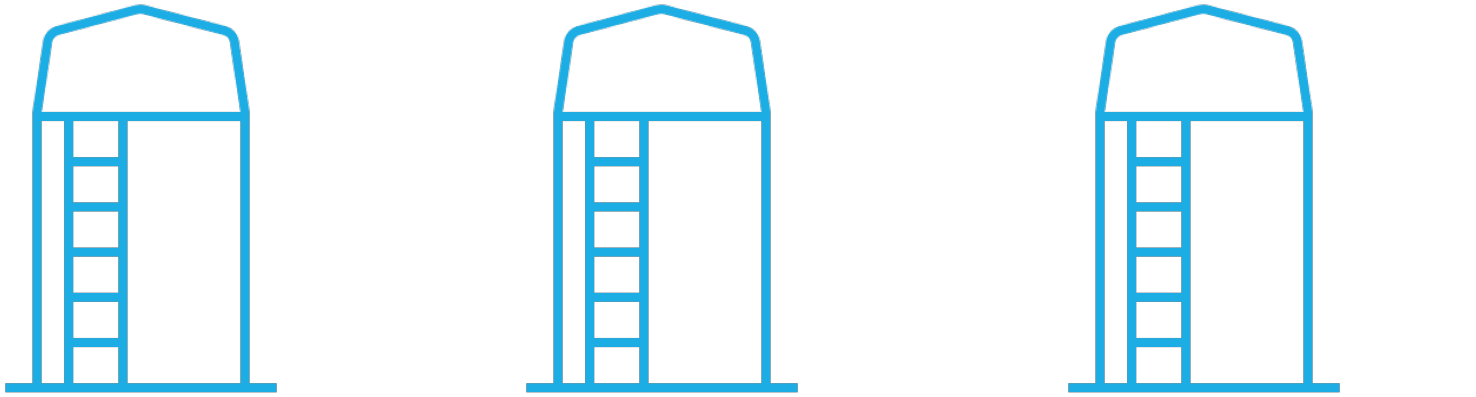
- IDEs allow developers to begin work quickly and efficiently due to all the tools that come out of the box
- IDEs also parse all code, and helps find bugs before they are committed by human error
- The GUIs of IDEs is created with one goal in mind, help developers be efficient.

SETTING UP A PYTHON IDE

PYTHON VIRTUAL ENVIRONMENTS

PYTHON VIRTUAL ENVIRONMENT

- A virtual environment is a Python environment that is isolated from those in other Python environments.



HOW ARE WE GOING TO INSTALL THE DEPENDENCIES?

- Pip: Pip is the Python package manager.
  - Pip is used to install and update packages.
  - You will want to make sure you have the latest version of pip installed

Unix/Mac OS

```
python3 -m pip --version
# pip 21.2.4
```

Windows

```
python -m pip --version
# pip 21.2.4
```

HOW WILL WE BE SETTING UP THE VIRTUAL ENVIRONMENT

- I started by creating a brand-new folder / directory called “FastAPI”
- Within our pip we will be checking what dependencies we already have installed.
- Installing Virtual Env if it is not installed.
- Creating a new FastAPI environment as a virtual environment
- Activating our virtual environment
- Lastly, installing FastAPI to our virtual environment

SWAGGER, OPENAPI, REQUEST METHODS & STATUS CODES

OPENAPI SPECIFICATION (OAS):

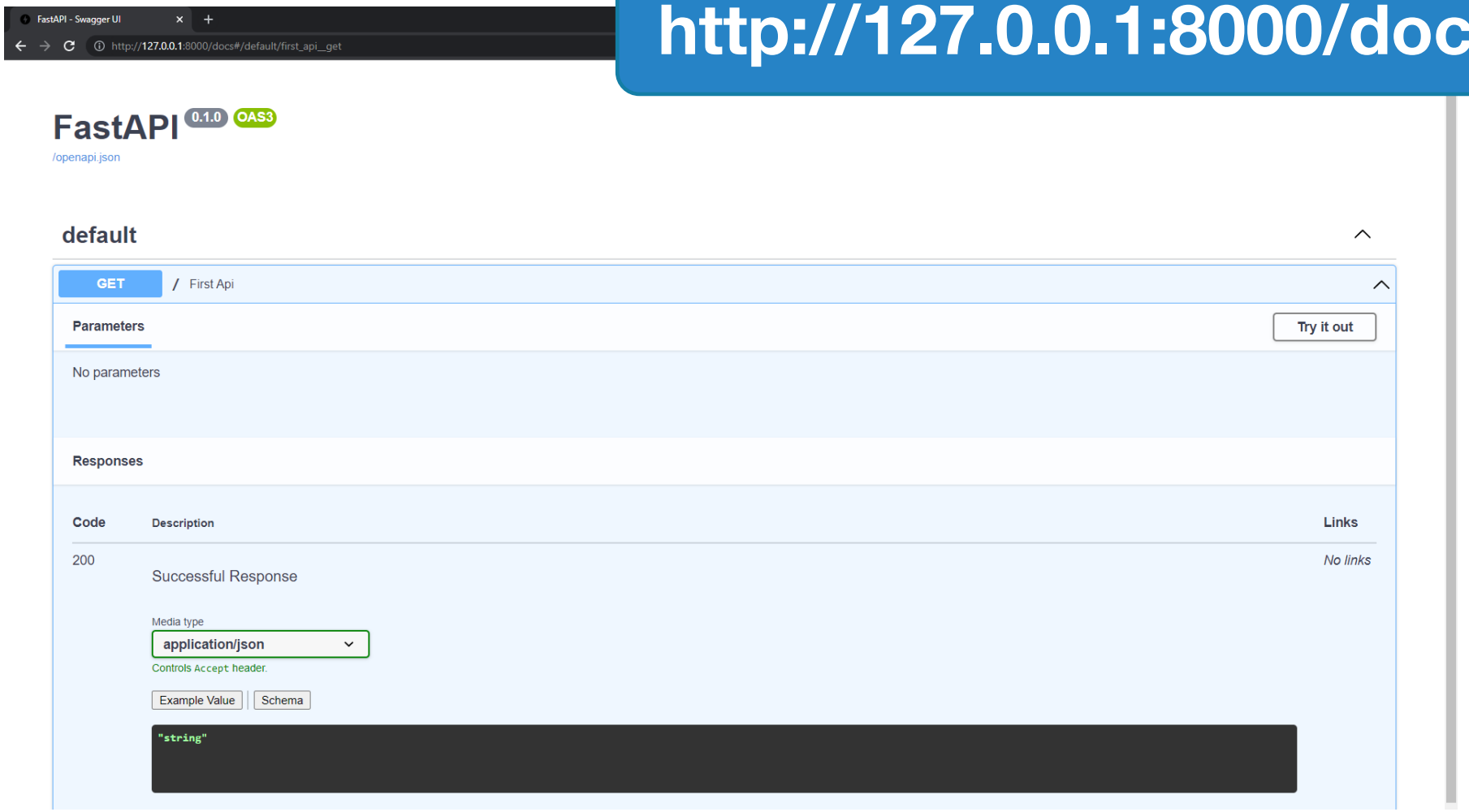
- OpenAPI Document Defines:
  - Schema
  - Data Format
  - Data Type
  - Path
  - Object
  - And much more

VIEW OPENAPI SCHEMA

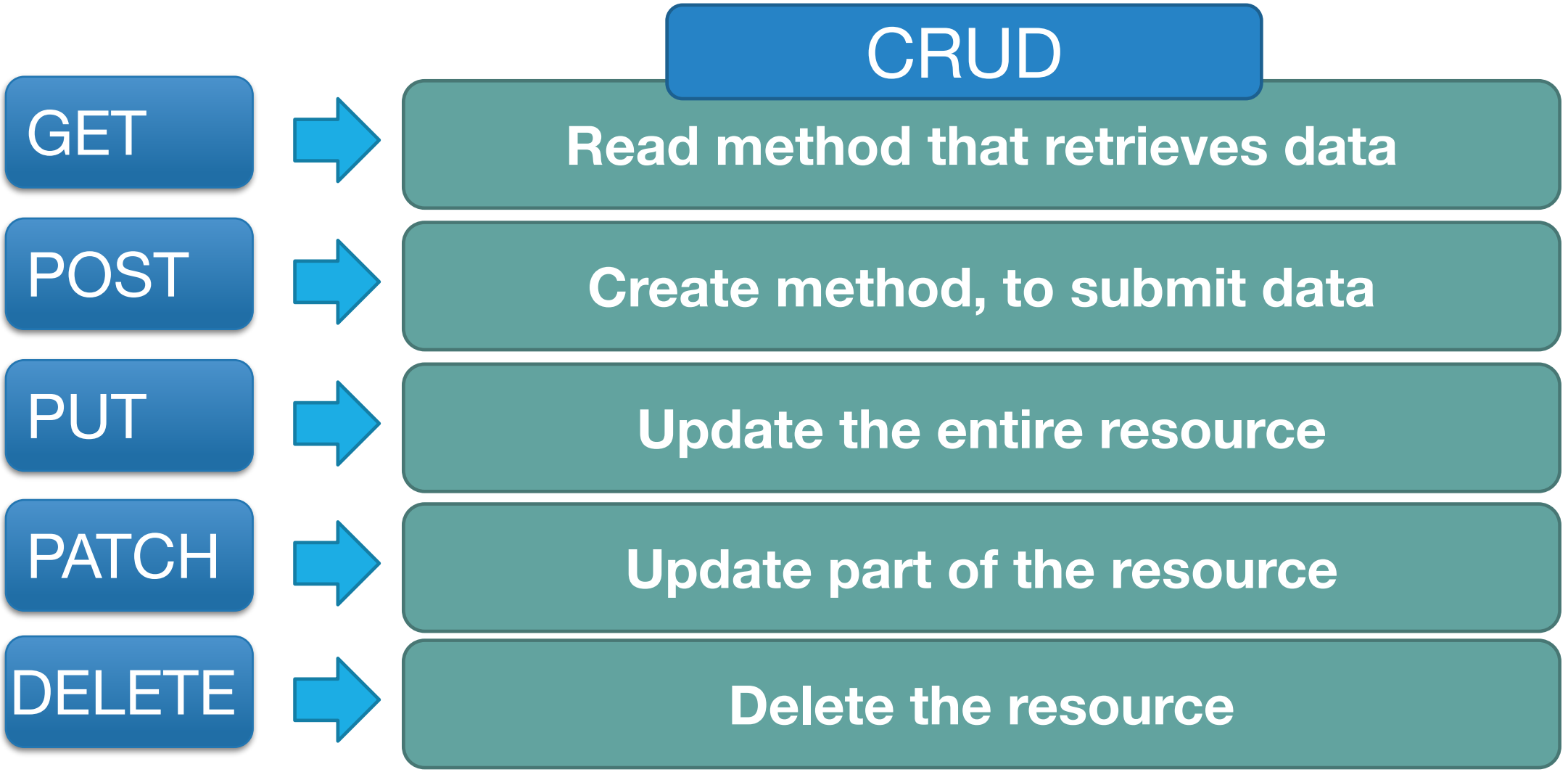
- FastAPI generates the OpenAPI schema so you can view
- http://127.0.0.1:8000/openapi.json
- Helps the developer create RESTful APIs based on standards so individuals can use the APIs easily

SWAGGER-UI

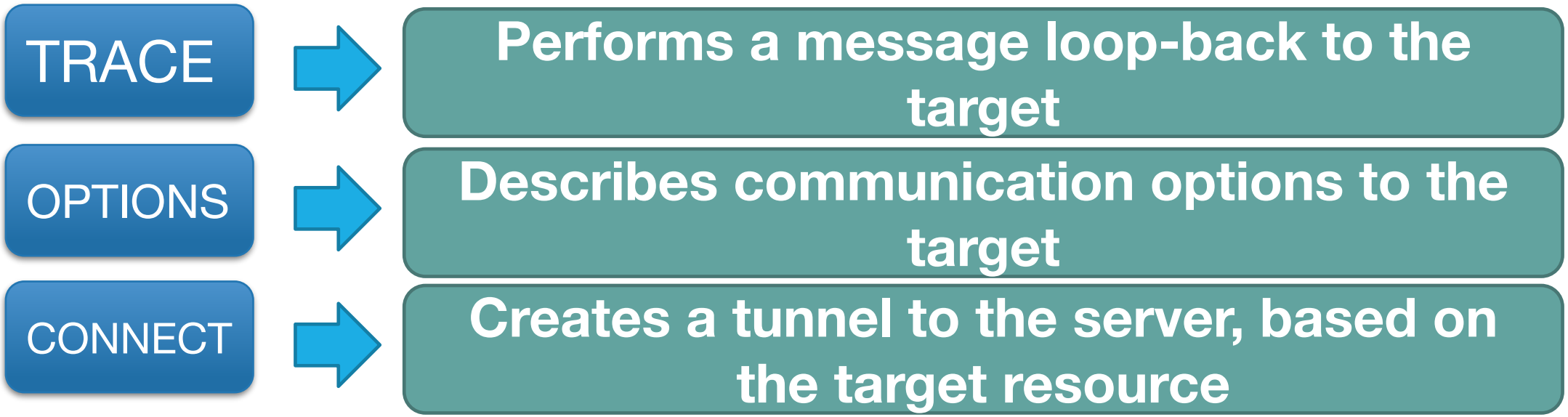
http://127.0.0.1:8000/docs

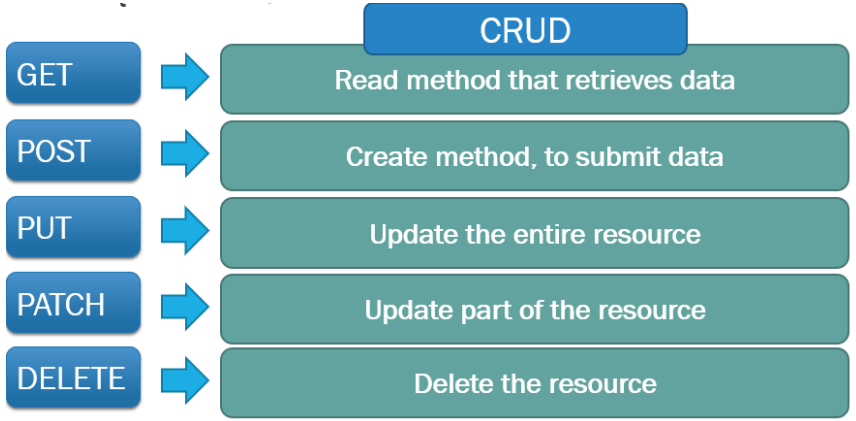


FASTAPI USES HTTP REQUEST METHODS

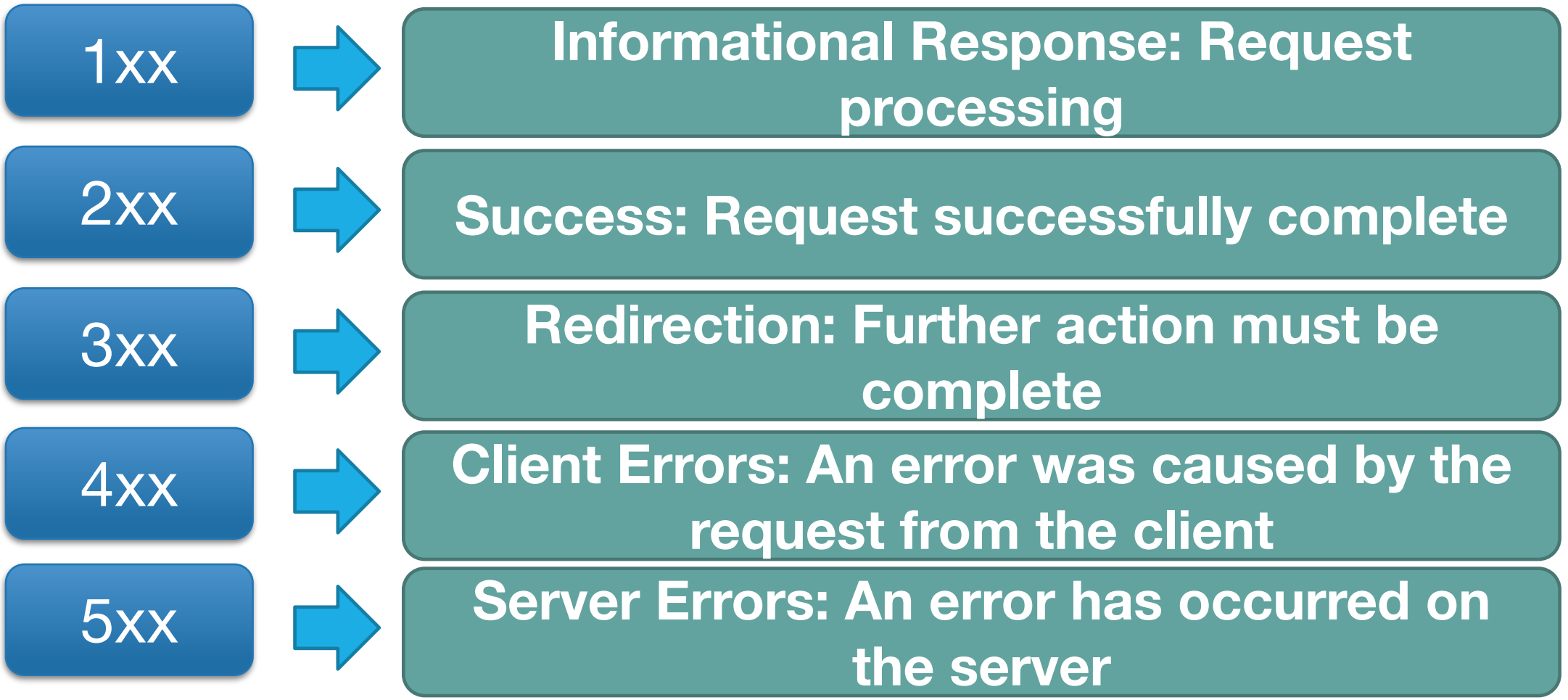


FASTAPI REQUEST METHODS (CONT)

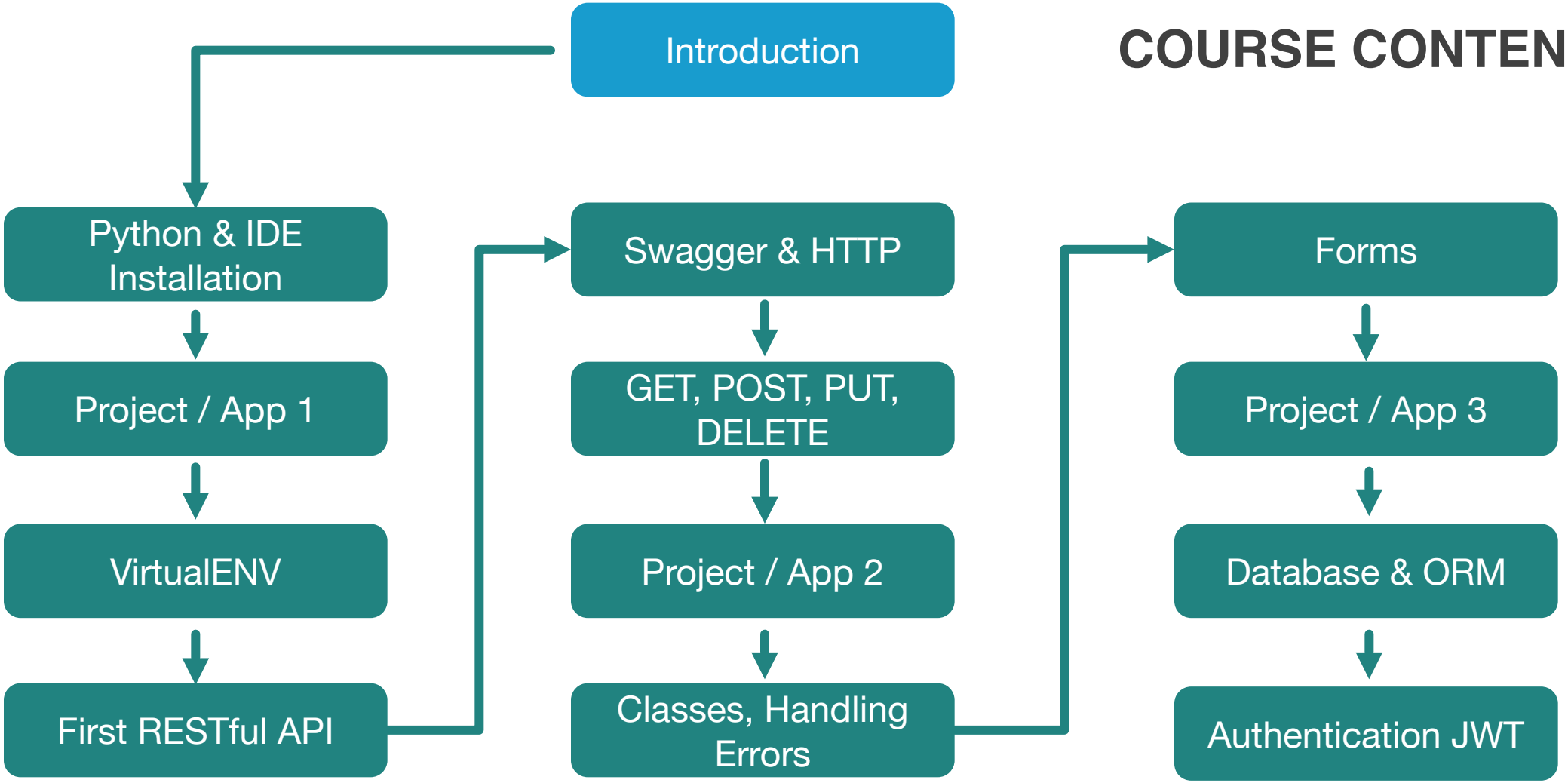


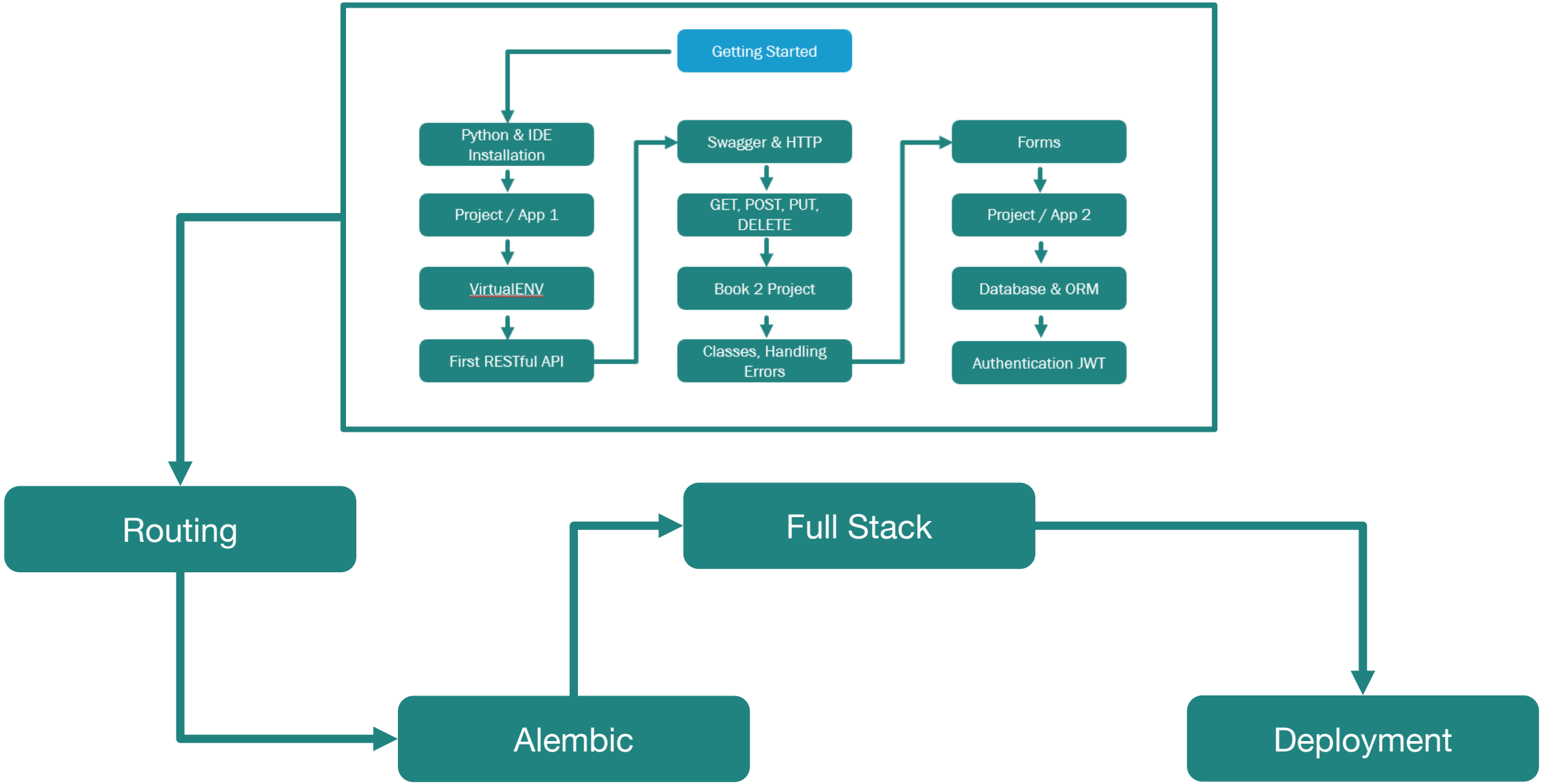


FASTAPI RESPONSE STATUS CODE



COURSE CONTENT!





HOW TO GET THE MOST OUT OF THE COURSE!

Watch all the Videos	Control the speed of videos to your pace
Take the Exercises	Pause videos to take exercises before seeing solutions
Search & Practice	Advance on your own and search for additional information
Ask Questions	Ask in Q&A section!

FASTAPI OVERVIEW

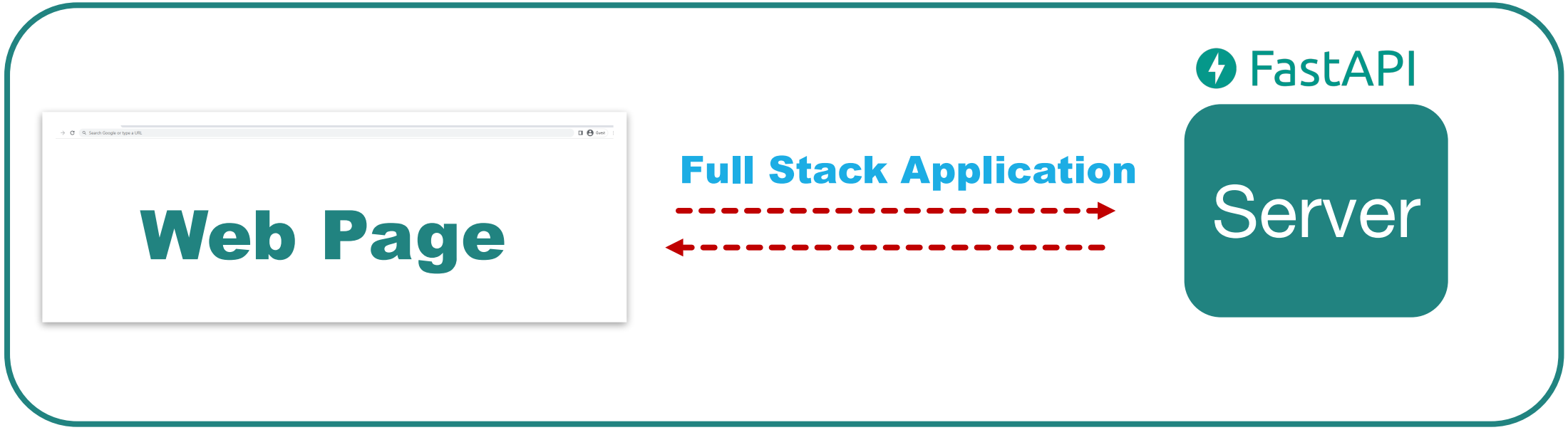
WHAT IS FASTAPI?

- FastAPI is a Python web-framework for building modern APIs
  - Fast (Performance)
  - Fast (Development)
- Key Notes:
  - Few Bugs
  - Quick & Easy
  - Robust
  - Standards

WHAT DOES THIS ALL MEAN FOR YOU?

- FastAPI is a web-framework for building modern RESTful APIs

WHERE DOES FASTAPI FIT WITHIN AN APPLICATION ARCHITECTURE?

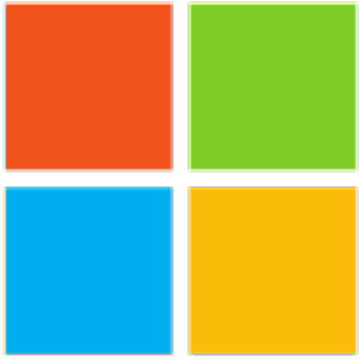


Handles all business logic for the application

WHO HAS USED FASTAPI?

Netflix

Uber



Microsoft

CAN'T I JUST WRITE EVERYTHING MYSELF?

- FAQ: Why do I need a web framework?
  - You may be able to write everything yourself, but why reinvent the wheel?
  - Web-frameworks allow a simplified way for rapid development.
  - Includes many years of development, which allows you to have a secure and fast application! ☺

POSTMAN OVERVIEW

WHAT IS POSTMAN

- Postman is an API software and platform for using APIs
- Postman is the industry standard for testing APIs
- Postman comes with many tools for design and testing of APIs, whether the API is built using SOAP, REST or another technology

POSTMAN'S API CLIENT

- Postman has a wonderful API Client that enables you to easily design, explore and debug APIs
- An API Client allows you to completely fulfill an API request using headers for things like authentication
- Postman allows you to send API requests to HTTP, SOAP, REST, GraphQL and WebSockets

POSTMAN API DESIGN

- Postman can help you with API design
- Can create multiple different stages of an API to track lifecycle for:
  - Documentation
  - Tests
  - And more!

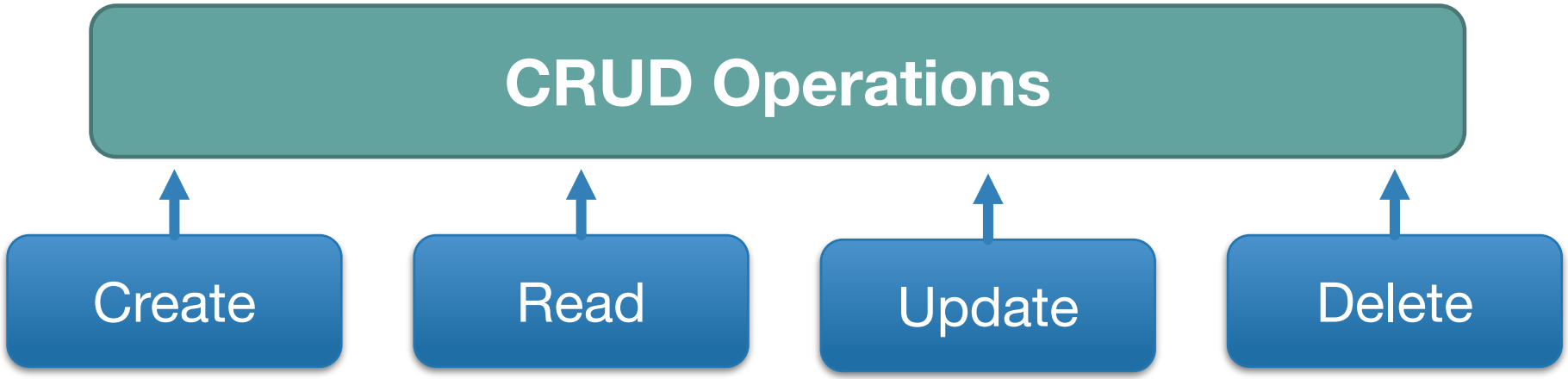
POSTMAN HAS MANY FEATURES

- Postman also allows you to do much more:
  - Create Mock Servers
  - Monitor your APIs
  - Create private and public networks
  - Reporting
  - Security alerts and more!
- Lets dive into Postman!

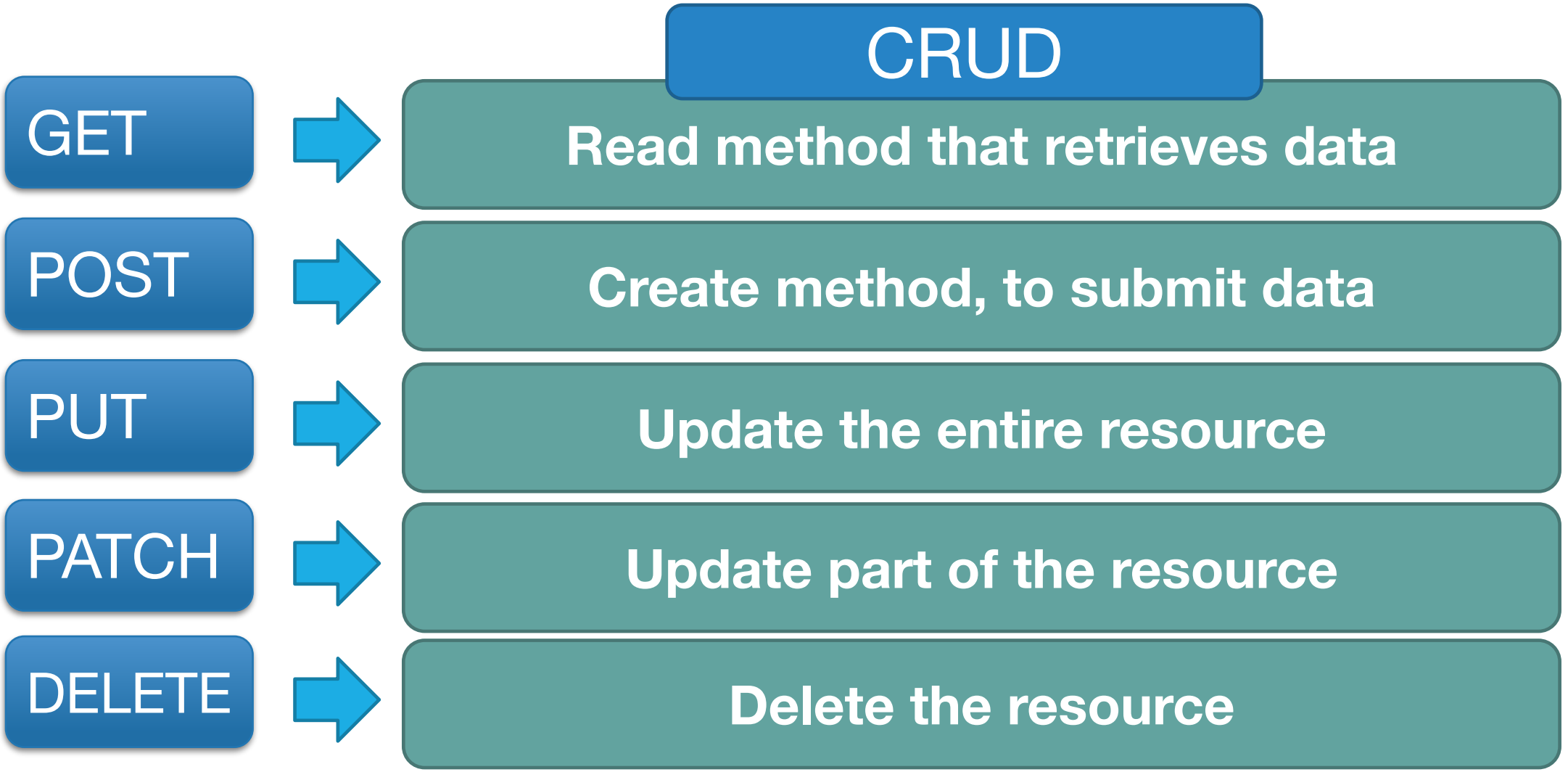
BOOKS PROJECT

WHAT WILL WE BE CREATING?

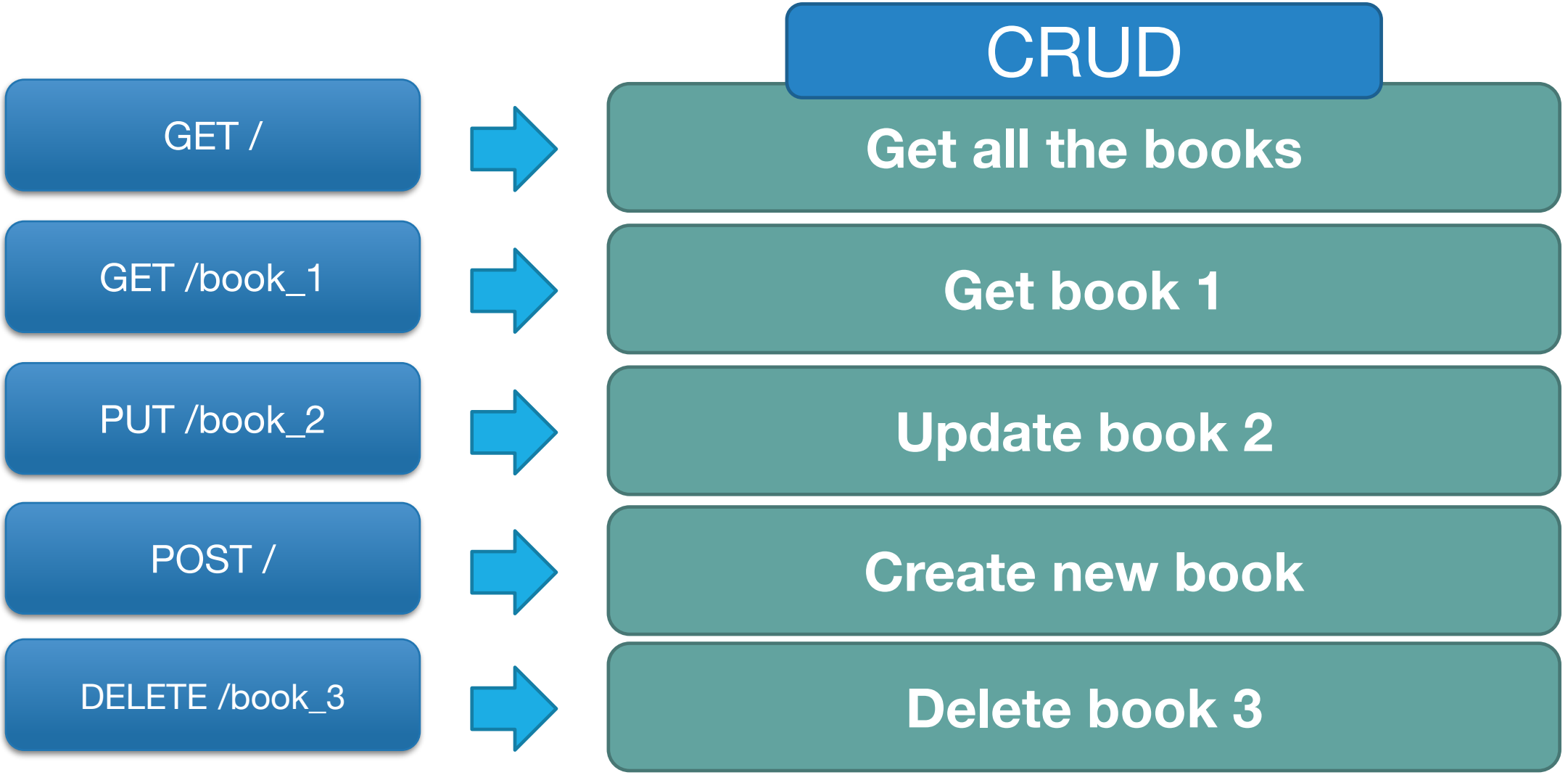
```
BOOKS = {
    'book_1': {'title': 'Title One', 'author': 'Author One'},
    'book_2': {'title': 'Title Two', 'author': 'Author Two'},
    'book_3': {'title': 'Title Three', 'author': 'Author Three'},
    'book_4': {'title': 'Title Four', 'author': 'Author Four'},
    'book_5': {'title': 'Title Five', 'author': 'Author Five'},
}
```



REST USES HTTP REQUEST METHODS

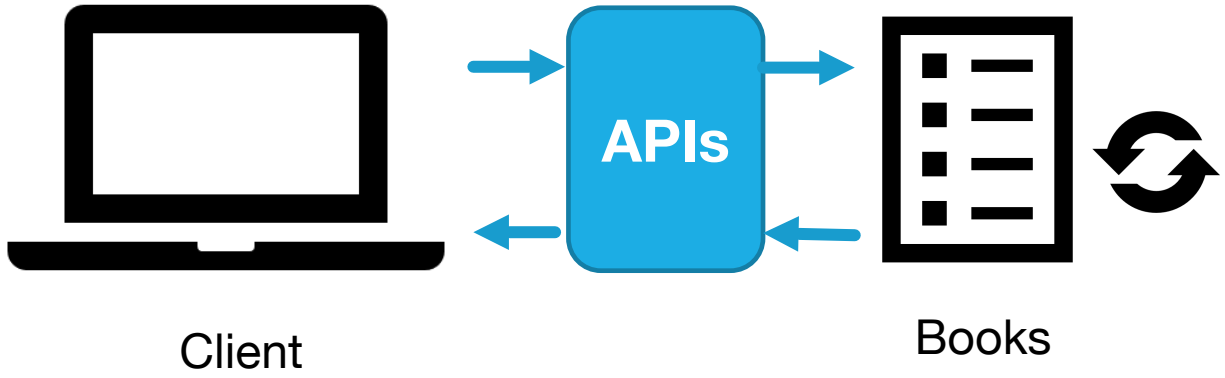


REST USES HTTP REQUEST METHODS



HOW WILL THIS LOOK

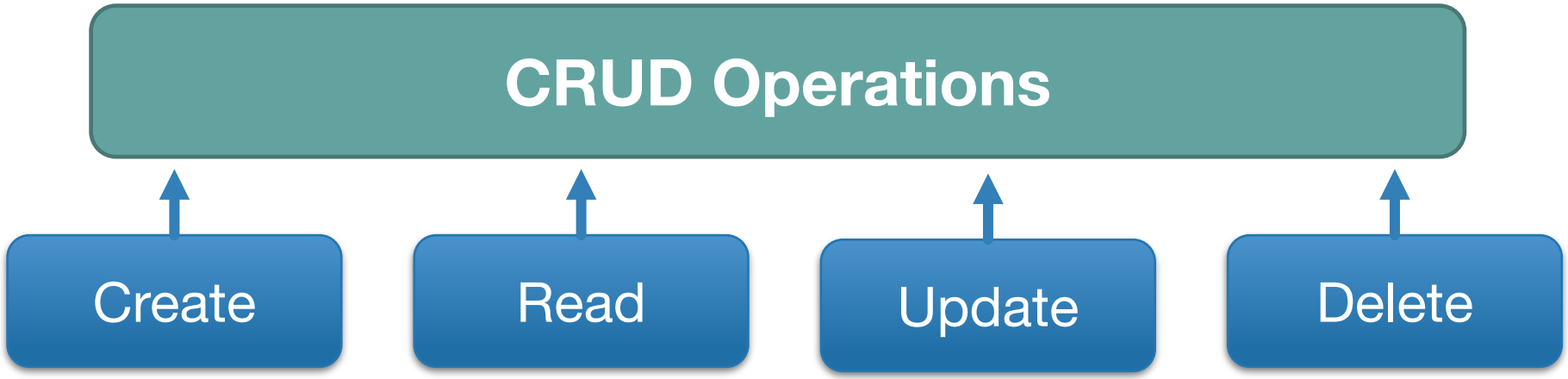
- We will first create the APIs on FastAPI
- Test the APIs on Swagger UI Documentation
- We will be able to see APIs working that we built!



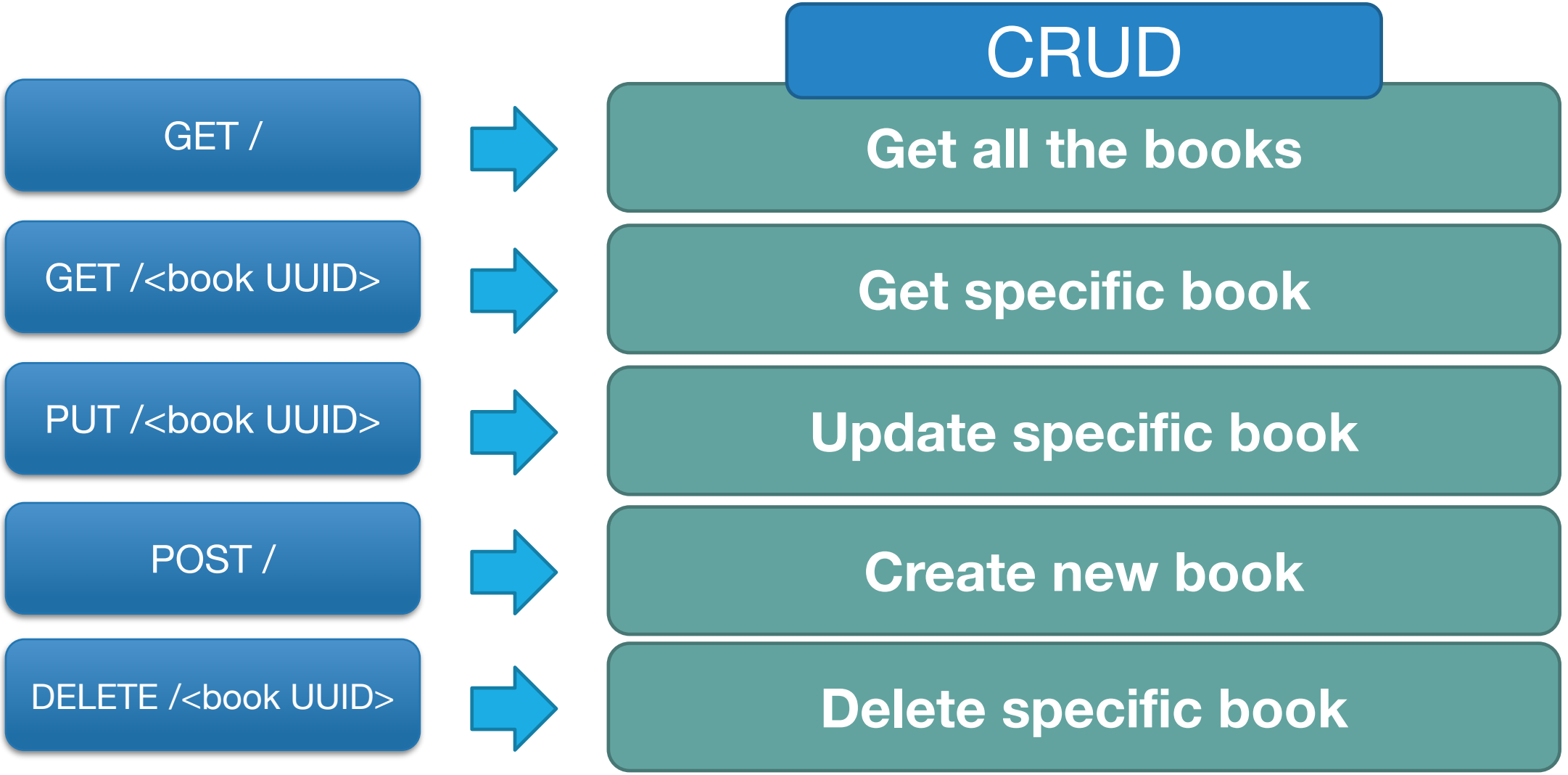
BOOKS 2 PROJECT

WHAT WILL WE BE CREATING?

```python
class Book(BaseModel):
    id: UUID
    title: str = Field(min_length=1)
    author: str = Field(min_length=1, max_length=100)
    description: Optional[str] = Field(title="Description of the book",
                                       max_length=100,
                                       min_length=1)
    rating: int = Field(gt=-1, lt=101)
```



REST USES HTTP REQUEST METHODS



TODOS PROJECT

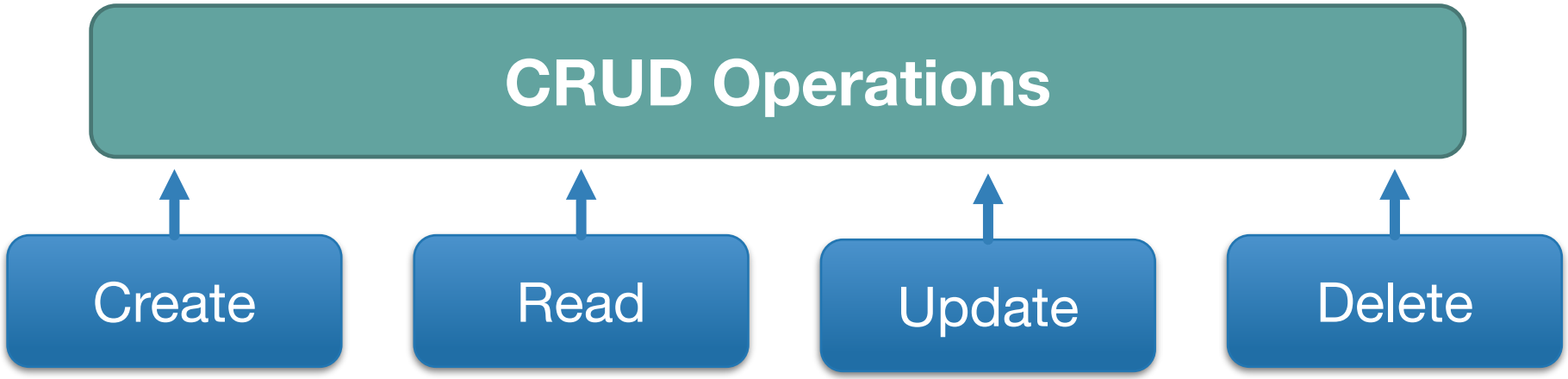
TODO PROJECT HAS MANY FEATURES

- Todo Project
  - Create APIs using Request Methods
  - Full SQL Database to save todos
  - Authentication using Bcrypt hashed password
  - Authorization using JSON Web Tokens (JWT)

```python
class Users(Base):
    __tablename__ = "users"

    id = Column(Integer, primary_key=True, index=True)
    email = Column(String, unique=True, index=True)
    username = Column(String, unique=True, index=True)
    first_name = Column(String)
    last_name = Column(String)
    hashed_password = Column(String)
    is_active = Column(Boolean, default=True)

    todos = relationship("Todos", back_populates="owner")


class Todos(Base):
    __tablename__ = "todos"

    id = Column(Integer, primary_key=True, index=True)
    title = Column(String)
    description = Column(String)
    priority = Column(Integer)
    complete = Column(Boolean, default=False)
    owner_id = Column(Integer, ForeignKey("users.id"))

    owner = relationship("Users", back_populates="todos")
```

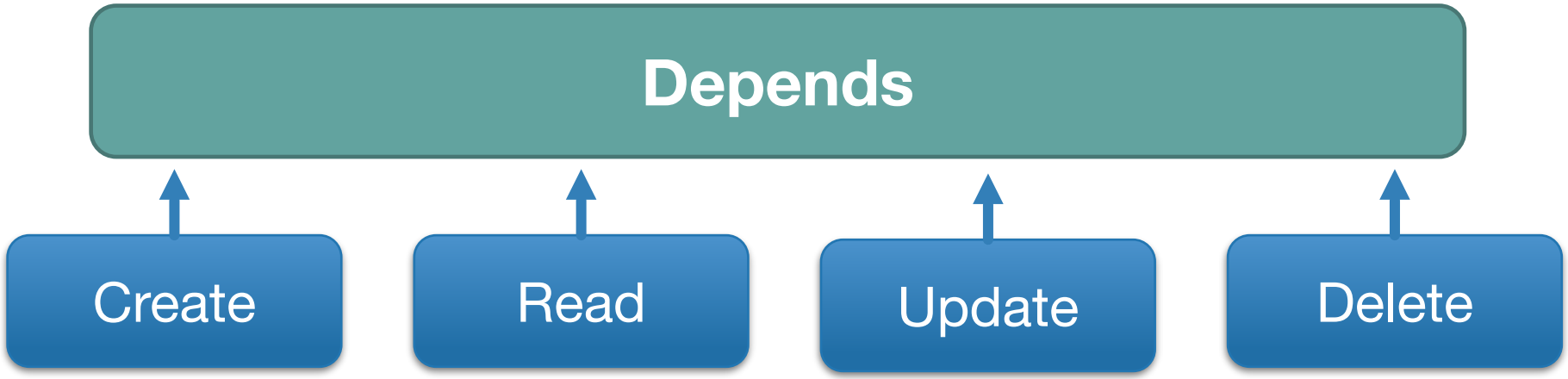
WHAT WILL WE BE CREATING?

```python
class Todo(BaseModel):
    title: str
    description: Optional[str]
    priority: int = Field(gt=0, lt=6, description="The priority must be between 1-5")
    complete: bool

class CreateUser(BaseModel):
    username: str
    email: Optional[str]
    first_name: str
    last_name: str
    password: str
```

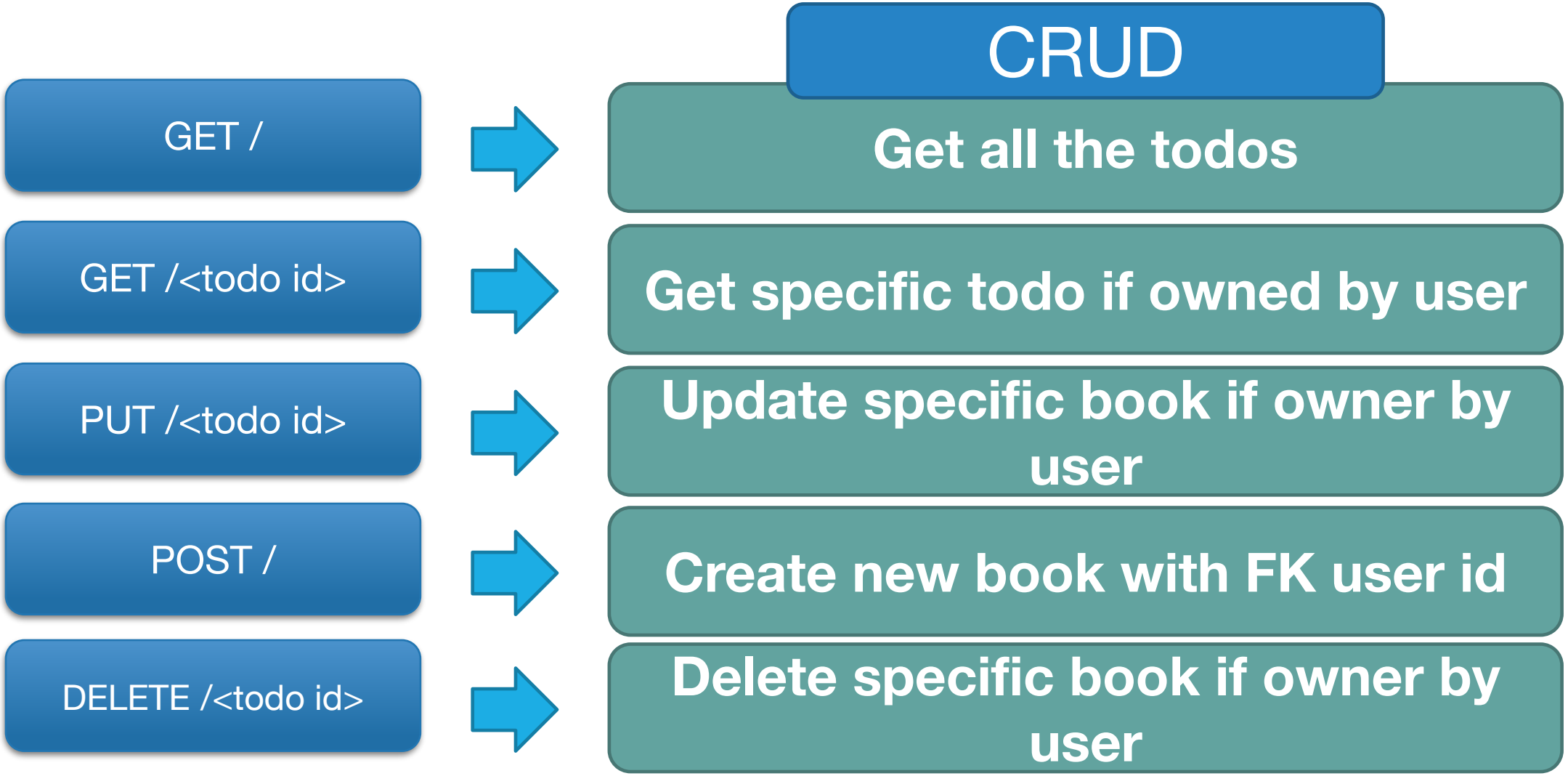


WHAT WILL WE BE CREATING?

```python
@app.post("/token")
async def login_for_access_token(form_data: OAuth2PasswordRequestForm = Depends(),
                                 db: Session = Depends(get_db)):
    user = authenticate_user(form_data.username, form_data.password, db)
    if not user:
        raise token_exception()
    token_expires = timedelta(minutes=20)
    token = create_access_token(user.username,
                                user.id,
                                expires_delta=token_expires)
    return {"token": token}
```



REST USES HTTP REQUEST METHODS



JSON WEB TOKEN (JWT) OVERVIEW

WHAT IS A JSON WEB TOKEN?

- JSON Web Token is a self-contained way to securely transmit data and information between two parties using a JSON Object.
- JSON Web Tokens can be trusted because each JWT can be digitally signed, which in return allows the server to know if the JWT has been changed at all
- JWT should be used when dealing with authorization
- JWT is a great way for information to be exchanged between the server and client

JSON WEB TOKEN STRUCTURE

- A JSON Web Token is created of three separate parts separated by dots (.) which include:
  - Header : (a)
  - Payload : (b)
  - Signature : (c)

JWT HEADER

- A JWT header usually consist of two parts:
  - (alg) The algorithm for signing
  - “typ” The specific type of token
- The JWT header is then encoded using Base64 to create the first part of the JWT (a)

JWT PAYLOAD

- A JWT Payload consists of the data. The Payloads data contains claims, and there are three different types of claims.
  - Registered
  - Public
  - Private
- The JWT Payload is then encoded using Base64 to create the second part of the JWT (b)

```
{
  "sub": "1234567890",
  "name": "Eric Roby",
  "given_name": "Eric",
  "family_name": "Roby",
  "email": "codingwithroby@gmail.com"
  "admin": true
}
```

```
aaaaaaaa.bbbbbbbb.cccccccc
```

JWT SIGNATURE

- A JWT Signature is created by using the algorithm in the header to hash out the encoded header, encoded payload with a secret.
- The secret can be anything, but is saved somewhere on the server that the client does not have access to
- The signature is the third and final part of a JWT (c)

```
HMACSHA256(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
  secret)
```

```
aaaaaaaa.bbbbbbbb.cccccccc
```

JWT EXAMPLE

JWT Header

```
{
  "alg": "HS256",
  "typ": "JWT"
}
```

JWT Payload

```
{
  "sub": "1234567890",
  "name": "Eric Roby",
  "given_name": "Eric",
  "family_name": "Roby",
  "email": "codingwithroby@gmail.com"
  "admin": true
}
```

JWT Signature

```
HMACSHA256(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
  learnonline)
```

JSON Web Token

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkVyaWMgUm9ieSIsImdpdmVuX25hbWUiOiJFcmljIiwiZmFtaWx5X25hbWUiOiJSb2J5IiwiZW1haWwiOiJjb2Rpbmd3aXRocm9ieUBnbWFpbC5jb20iLCJhZG1pbiI6dHJ1ZX0.i8yqzOz-nWr5hwXqRYP18W9igUPoKMZiBZW315tK5g8
```

Encoded PASTE A TOKEN HERE

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.ey
JzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkVya
WMgUm9ieSIsImdpdmVuX25hbWUiOiJFcmljIiwi
ZmFtaWx5X25hbWUiOiJSb2J5IiwiZW1haWwiOiJ
jb2RpbmdpdGhyb2J5QGdtYWlsLmNvbSIsImFkbW
1pbiI6dHJ1ZX0.i8yqzOz-
nWr5hwXqRYP18W9igUPoKMZiBZW315tK5g8
```

Decoded EDIT THE PAYLOAD AND SECRET

https://jwt.io

HEADER: ALGORITHM & TOKEN TYPE

```
{
  "alg": "HS256",
  "typ": "JWT"
}
```

PAYLOAD: DATA

```
{
  "sub": "1234567890",
  "name": "Eric Roby",
  "given_name": "Eric",
  "family_name": "Roby",
  "email": "codingwithroby@gmail.com",
  "admin": true
}
```

VERIFY SIGNATURE

```
HMACSHA256(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
  learnonline
) secret base64 encoded
```

Signature Verified

SHARE JWT

Encoded PASTE A TOKEN HERE

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.ey
JzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkVya
WMgUm9ieSIsImdpdmVuX25hbWUiOiJFcmljIiwi
ZmFtaWx5X25hbWUiOiJSb2J5IiwiZW1haWwiOiJ
jb2RpbmdpdGhyb2J5QGdtYWlsLmNvbSIsImFkbW
1pbiI6dHJ1ZX0.i8yqzOz-
nWr5hwXqRYP18W9igUPoKMZiBZW315tK5g
```

Decoded EDIT THE PAYLOAD AND SECRET

https://jwt.io

HEADER: ALGORITHM & TOKEN TYPE

```
{
  "alg": "HS256",
  "typ": "JWT"
}
```

PAYLOAD: DATA

```
{
  "sub": "1234567890",
  "name": "Eric Roby",
  "given_name": "Eric",
  "family_name": "Roby",
  "email": "codingwithroby@gmail.com",
  "admin": true
}
```

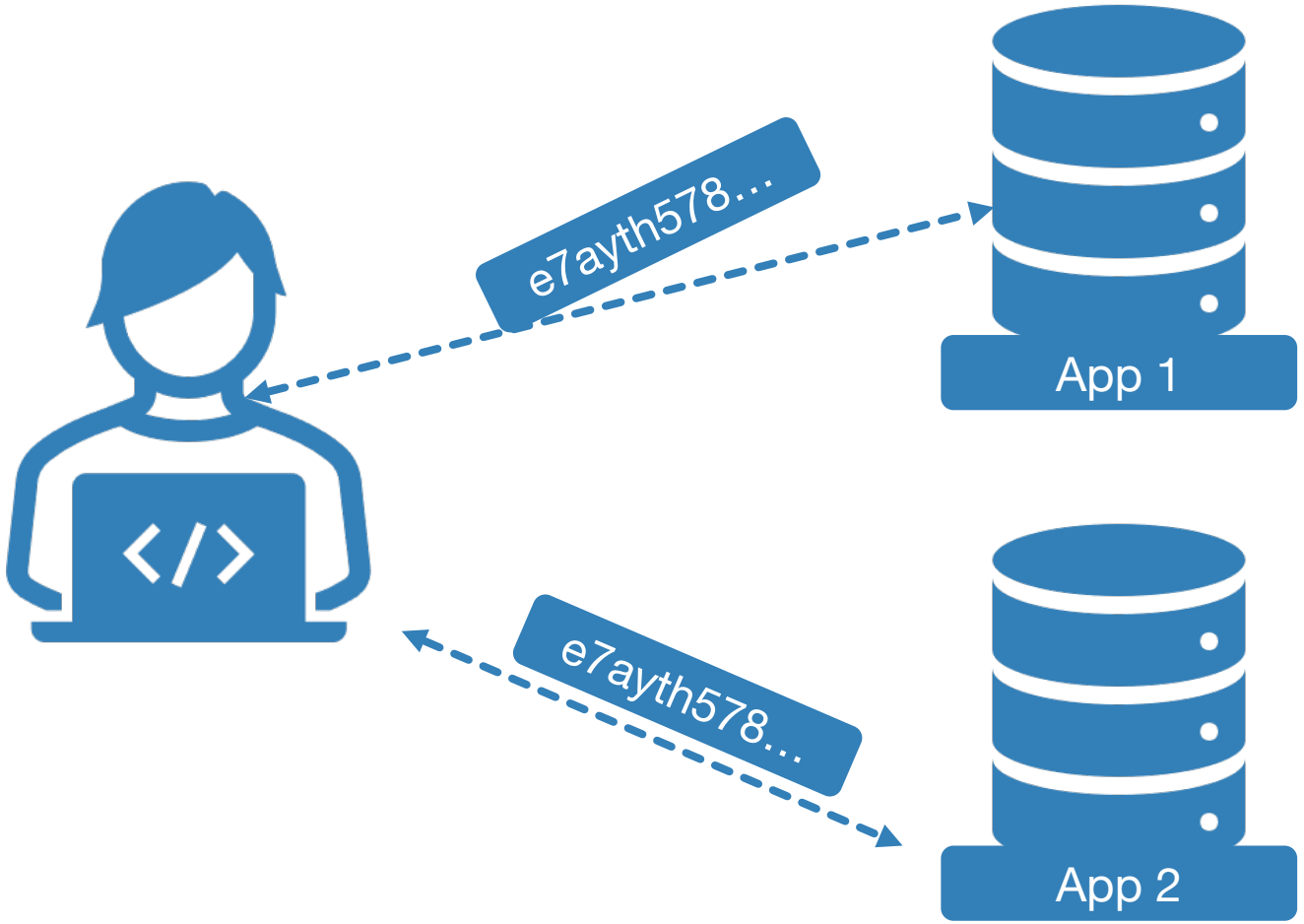
VERIFY SIGNATURE

```
HMACSHA256(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
  learnonline
) secret base64 encoded
```

⊗ Invalid Signature

SHARE JWT

JWT PRACTICAL USE CASE



PRODUCTION DATABASE INTRODUCTION

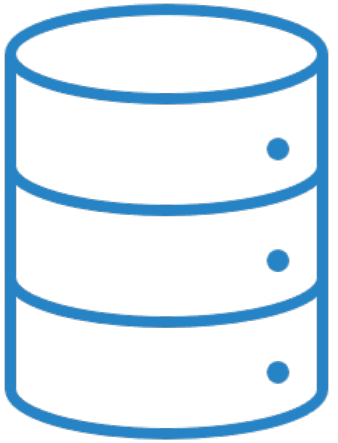
PRODUCTION DATABASE

- This section will go over installing a production
- either one

Before jumping in, lets take about the difference between SQLite and production DBMS

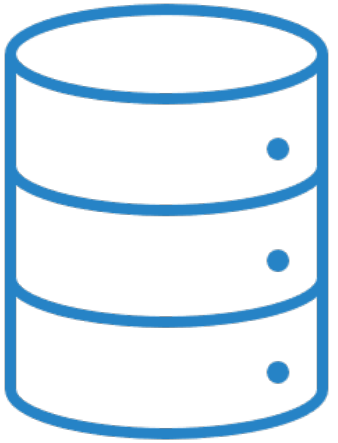
PRODUCTION DBMS VS SQLITE3

- SQLite3 strives to provide local data storage for individual applications and devices.
- SQLite3 emphasizes economy, efficiency and simplicity.
- For most small / medium applications, SQLite3 will works perfectly.
- SQLite3 focuses on different concepts than a production Database Management System.



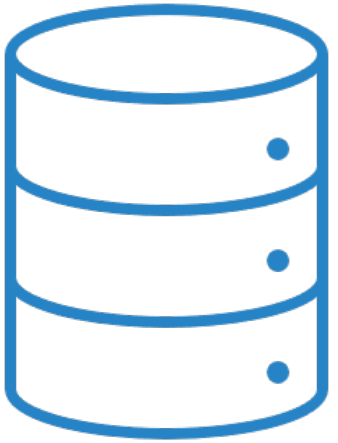
PRODUCTION DBMS VS SQLITE3

- MySQL & PostgreSQL focuses on a big difference compared to SQLite3.
- These production DBMS focuses on scalability, concurrency and control.
- If you application is going to have 10s of thousands of users, it may be wise to switch to a production DBMS
- If you application is only you, and a few others, SQLite3 will work great!



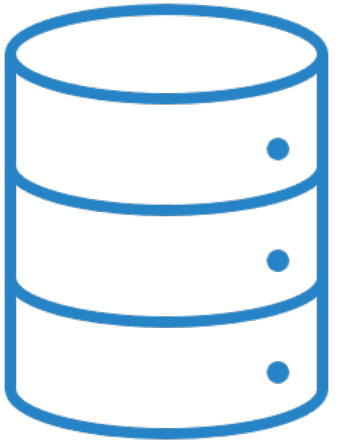
PRODUCTION DBMS KEY NOTES:

- SQLite3 runs in-memory, which allows development of a SQLite3 data to be easy, as it is part of your application!
- Production DBMS run on their own server and port. Which means you need to make sure the database is running, and have authentication linking to the DBMS
- (SQLite3) For deployment you can deploy a SQLite3 database along with the application
- (Prod DBMS) For deployment you will need to also deploy the database separate from the application



OVERVIEW OF SECTION (FOR BOTH MYSQL & POSTGRESQL)

- We will install the production DBMS
- Setup the tables and data within the production DBMS
- Connect the production DBMS to our application
- Push data from application to our production DBMS!

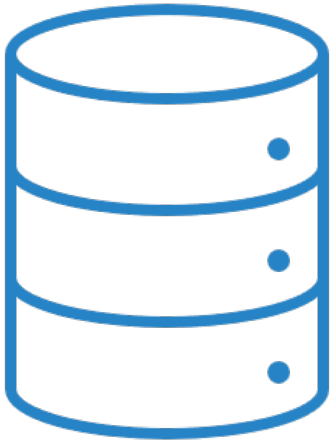


SQL PRODUCTION DATABASE INTRODUCTION

WHAT IS A DATABASE?

- Organized collection of structured information of **data**, which is stored in a computer system.
- The data
- The data
- The data
- Many databases use a structured query language (SQL) to modifying and writing data

What is **DATA**?



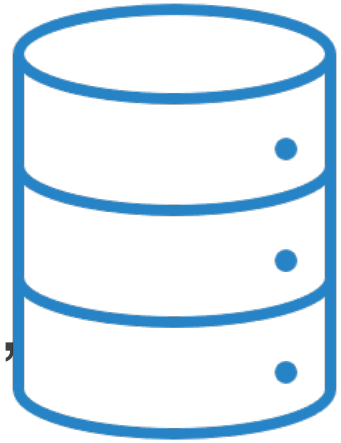
WHAT IS A DATA?

- Data can be related to just about any object
- For example, a user on an application may have:
  - Name
  - Age
  - Email
  - Gender
  - Password

All of this can be considered **data**

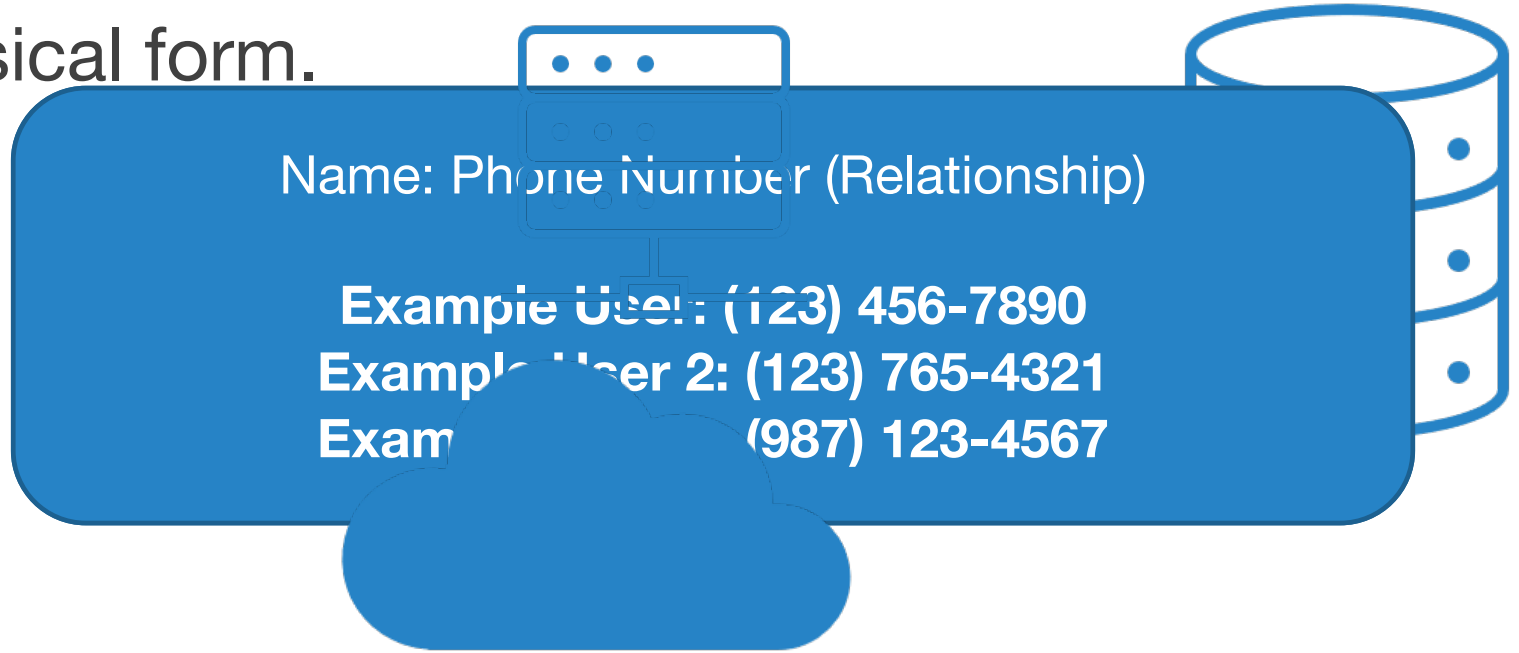
WHAT IS A DATABASE?

- A database is a collection of data.
- Since data, on its own, is just data. A database allows management to this data.
- Databases are organized in how data can be retrieved, stored and modified.
- There are many types of Database Management Systems (DBMS)



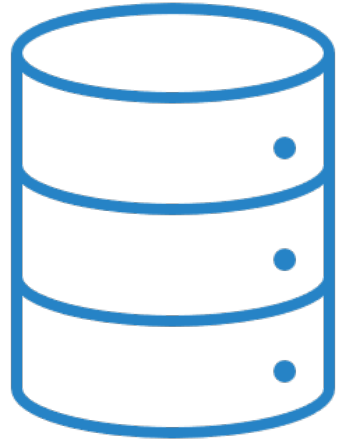
EXAMPLE OF A PHYSICAL DATABASE?

- Before we continue thinking electronically, lets think of a database in a physical form.
- Dictionary:
- Phone Book:



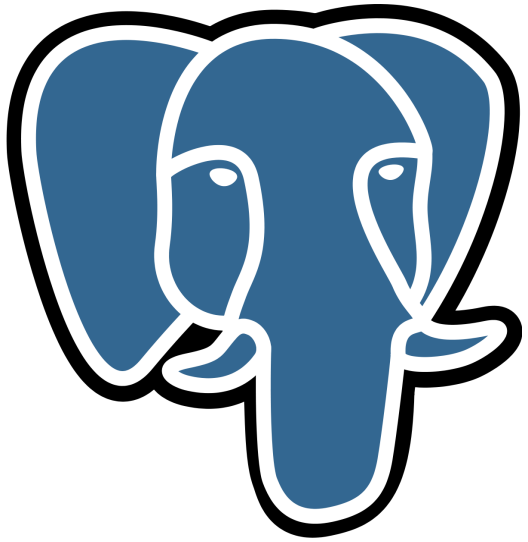
WHAT IS A SQL?

- Pronounced either as S-Q-L or “See Quel”
- Standard language for dealing with relational databases
- SQL can be used to:
  - Search
  - Update
  - Add
  - Delete
- Database Records



TOP SQL DBMS?

MySQL



PostgreSQL

SQLite

BASIC SQL QUERIES

INSERTING DATABASE TABLE (TODOS)

Id (PK)	title	description	priority	complete	owner (FK)

DELET THIS IS THE USERS TABLE

Id (PK)	email	username	first_name	last_name	hashed_password	is_active

DELET THIS IS THE USERS TABLE

Id (PK)	email	username	first_name	last_name	hashed_password	is_active
1	codingwithroby@gmail.com	codingwithroby	Eric	Roby	123abcEqw!..	1
2	exampleuser12@example.com	exampleuser	Example	User	Gjjjd!2!..	1

INSERTING DATABASE TABLE (TODOS)

```
INSERT INTO todos (title,
description, priority, complete)

VALUES (‘Go to store’, ‘To pick up
eggs’ 4, False);
```

Id (PK)	title	description	priority	complete
1	Go to store	To pick up eggs	4	0

INSERTING DATABASE TABLE (TODOS)

```
INSERT INTO todos (title,
description, priority, complete)

VALUES ('Haircut', 'Need to get
length 1mm' 3, False);
```

Id (PK)	title	description	priority	complete
1	Go to store	To pick up eggs	4	0
2	Haircut	Need to get length 1mm	3	0

STARTING DATABASE TABLE (TODOS)

Id (PK)	title	description	priority	complete
1	Go to store	To pick up eggs	4	0
2	Haircut	Need to get length 1mm	3	0

STARTING DATABASE TABLE (TODOS)

Id (PK)	title	description	priority	complete
1	Go to store	To pick up eggs	4	0
2	Haircut	Need to get length 1mm	3	0
3	Feed dog	Make sure to use new food brand	5	0

STARTING DATABASE TABLE (TODOS)

Id (PK)	title	description	priority	complete
1	Go to store	To pick up eggs	4	0
2	Haircut	Need to get length 1mm	3	0
3	Feed dog	Make sure to use new food brand	5	0
4	Water plant	Inside and Outside plants	4	0

STARTING DATABASE TABLE (TODOS)

Id (PK)	title	description	priority	complete
1	Go to store	To pick up eggs	4	0
2	Haircut	Need to get length 1mm	3	0
3	Feed dog	Make sure to use new food brand	5	0
4	Water plant	Inside and Outside plants	4	0
5	Learn something new	Learn to program	5	0

STARTING DATABASE TABLE (TODOS)

Id (PK)	title	description	priority	complete
1	Go to store	To pick up eggs	4	0
2	Haircut	Need to get length 1mm	3	0
3	Feed dog	Make sure to use new food brand	5	0
4	Water plant	Inside and Outside plants	4	0
5	Learn something new	Learn to program	5	0
6	Shower	You have not showered in days	5	0

DELETE

Id (PK)	title	description	priority	complete	owner
3	Feed dog	Make sure to use new food brand	5	0	2
4	Water plant	Inside and Outside plants	4	0	2
6	Shower	You have not showered in days	5	0	2

SELECT SQL QUERIES

```
SELECT * FROM todos;
```

Select ALL columns and rows

Id (PK)	title	description	priority	complete
1	Go to store	To pick up eggs	4	0
2	Haircut	Need to get length 1mm	3	0
3	Feed dog	Make sure to use new food brand	5	0
4	Water plant	Inside and Outside plants	4	0
5	Learn something new	Learn to program	5	0
6	Shower	You have not showered in days	5	0

SELECT SQL QUERIES

```
SELECT title FROM todos;
```

Select just title from columns

title
Go to store
Haircut
Feed dog
Water plant
Learn something new
Shower

SELECT SQL QUERIES

```
SELECT description FROM todos;
```

Select just description from columns

description
To pick up eggs
Need to get length 1mm
Make sure to use new food brand
Inside and Outside plants
Learn to program
You have not showered in days

SELECT SQL QUERIES

```
SELECT title, description FROM
todos;
```

Select title, description from columns

title	description
Go to store	To pick up eggs
Haircut	Need to get length 1mm
Feed dog	Make sure to use new food brand
Water plant	Inside and Outside plants
Learn something new	Learn to program
Shower	You have not showered in days

SELECT SQL QUERIES

```
SELECT title, description, priority FROM
todos;
```

Select title, description and priority from columns

title	description	priority
Go to store	To pick up eggs	4
Haircut	Need to get length 1mm	3
Feed dog	Make sure to use new food brand	5
Water plant	Inside and Outside plants	4
Learn something new	Learn to program	5
Shower	You have not showered in days	5

WHERE SQL QUERIES

WHERE SQL QUERIES

```
SELECT * FROM todos WHERE priority=5;
```

Select ALL rows & columns WHERE priority = 5

Id (PK)	title	description	priority	complete
3	Feed dog	Make sure to use new food brand	5	0
5	Learn something new	Learn to program	5	0
6	Shower	You have not showered in days	5	0

WHERE SQL QUERIES

```
SELECT * FROM todos WHERE title='Feed dog';
```

Select ALL rows & columns WHERE title= Feed dog

Id (PK)	title	description	priority	complete
3	Feed dog	Make sure to use new food brand	5	0

WHERE SQL QUERIES

```
SELECT * FROM todos WHERE id=2;
```

Select ALL rows & columns WHERE id= 2

Id (PK)	title	description	priority	complete
2	Haircut	Need to get length 1mm	3	0

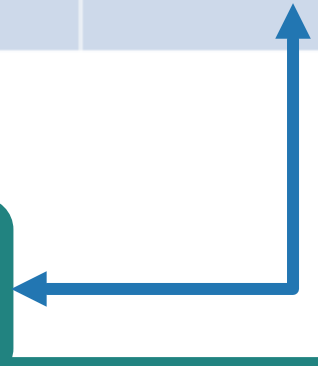
UPDATE SQL QUERIES

WHERE SQL QUERIES

```
UPDATE todos SET complete=True WHERE
title='Learn something new';
```

Update ALL rows & columns to now have complete = True
WHERE id = 5

Id (PK)	title	description	priority	complete
5	Learn something new	Learn to program	5	1



1 == True

WHERE SQL QUERIES

```
UPDATE todos SET complete=True WHERE id=5;
```

Update ALL rows & columns to now have complete = True
WHERE id = 5

Id (PK)	title	description	priority	complete
5	Learn something new	Learn to program	5	1

1 == True

DELETE SQL QUERIES

DELETE Clause

DELETE SQL QUERIES

```
DELETE FROM todos WHERE id=5;
```

Delete ALL rows and columns where id = 5

Id (PK)	title	description	priority	complete
1	Go to store	To pick up eggs	4	0
2	Haircut	Need to get length 1mm	3	0
3	Feed dog	Make sure to use new food brand	5	0
4	Water plant	Inside and Outside plants	4	0
6	Shower	You have not showered in days	5	0

DELETE SQL QUERIES

```
DELETE FROM todos WHERE complete=0;
```

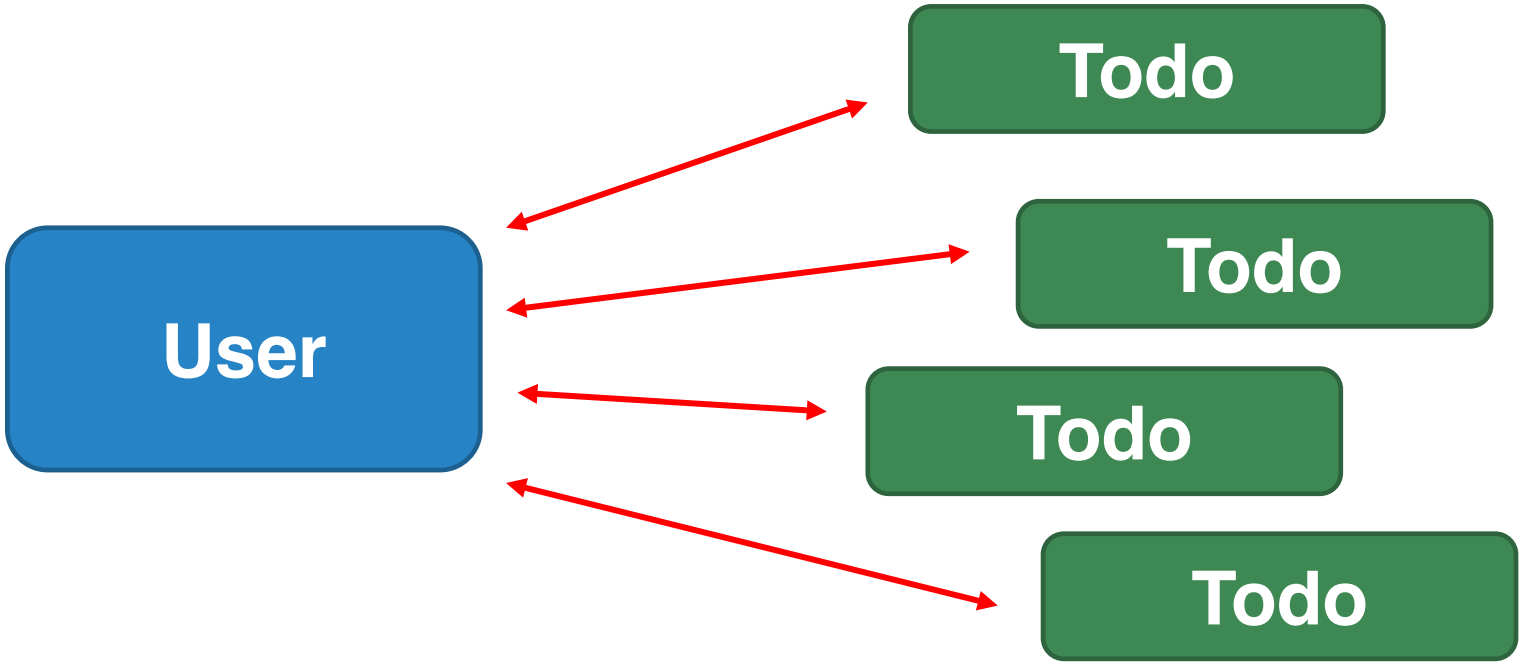
Delete ALL rows and columns where complete = 0

Id (PK)	title	description	priority	complete

ONE TO MANY INTRODUCTION

WHAT IS A ONE TO MANY RELATIONSHIP

- A user can have many todos



WHAT IS A ONE TO MANY RELATIONSHIP

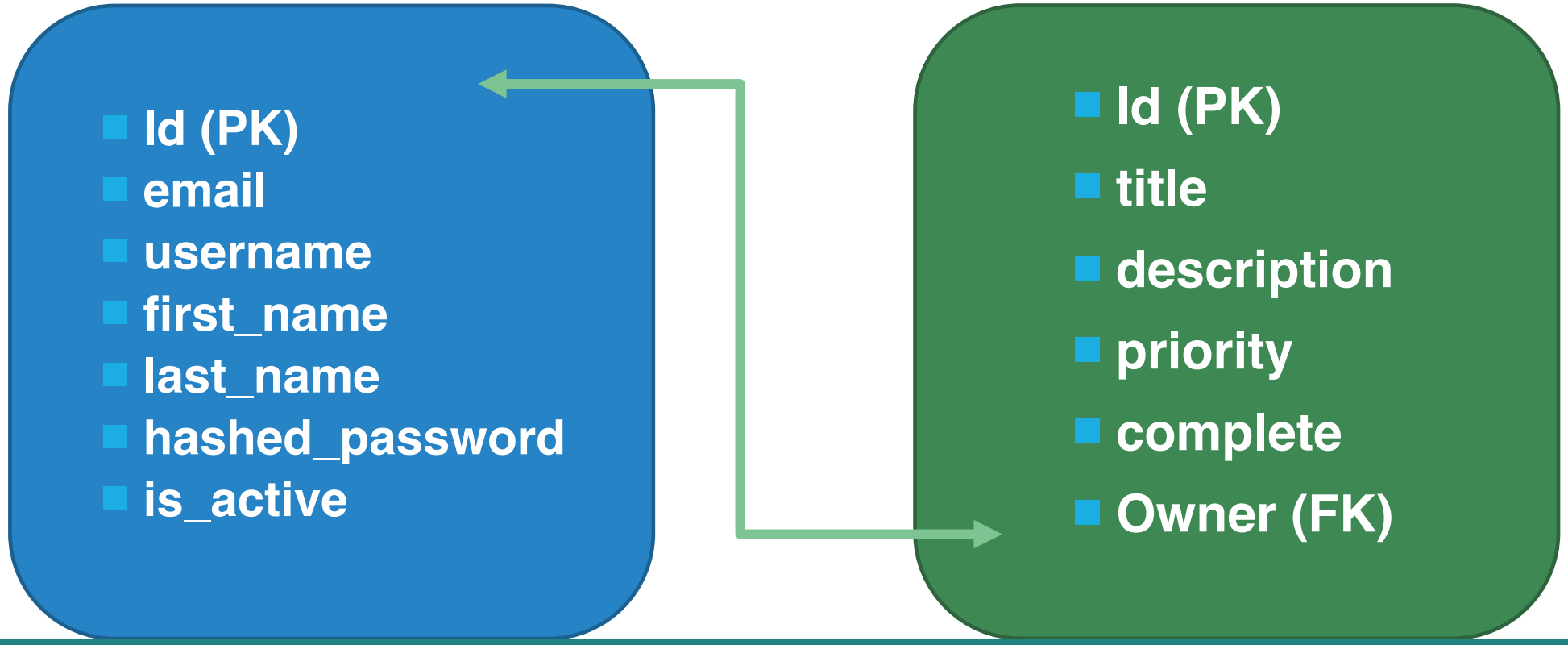
Users & Todos

Id (PK)	email	username	first_name	last_name	hashed_password	is_active

Id (PK)	title	description	priority	complete

WHAT IS A ONE TO MANY RELATIONSHIP

Users & Todos



WHAT IS A ONE TO MANY RELATIONSHIP

Id (PK)	title	description	priority	complete

Add new column

Id (PK)	title	description	priority	complete	owner (FK)

WHAT IS A ONE TO MANY RELATIONSHIP

Users

Id (PK)	email	username	first_name	last_name	hashed_password	is_active
1	codingwithroby@gmail.com	codingwithroby	Eric	Roby	123abcEqw!..	1
2	exampleuser12@example.com	exampleuser	Example	User	Gjjjd!2!..	1

Todos

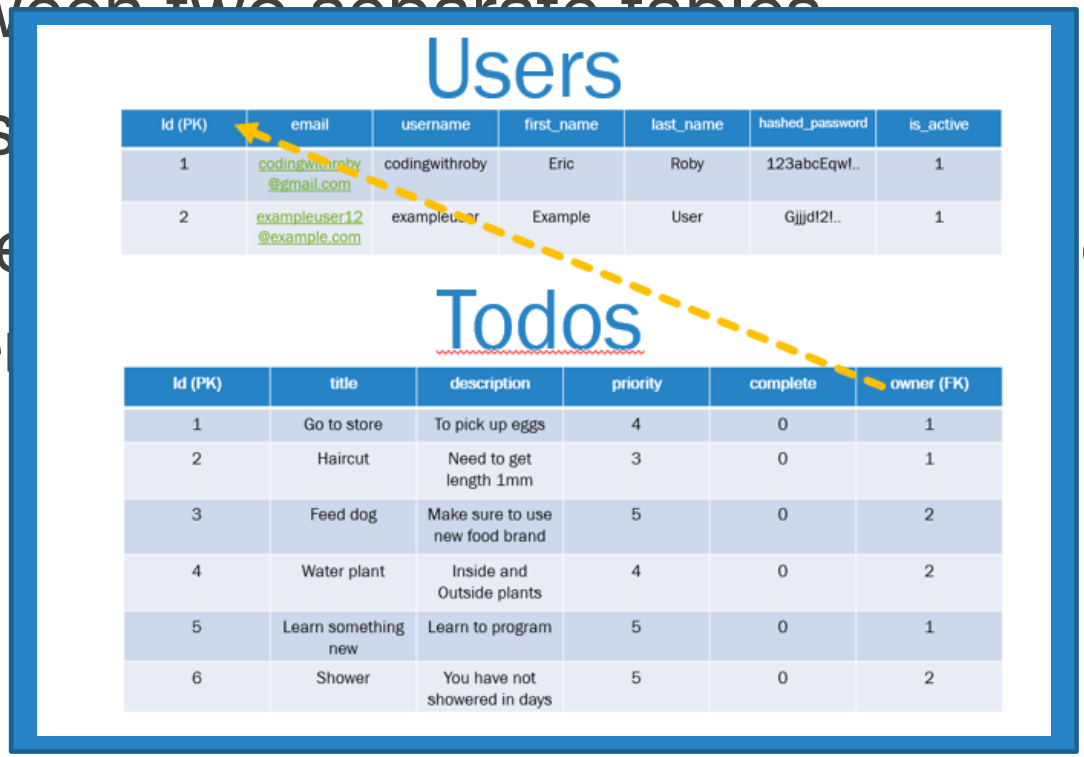
Id (PK)	title	description	priority	complete	owner (FK)
1	Go to store	To pick up eggs	4	0	1
2	Haircut	Need to get length 1mm	3	0	1
3	Feed dog	Make sure to use new food brand	5	0	2
4	Water plant	Inside and Outside plants	4	0	2
5	Learn something new	Learn to program	5	0	1
6	Shower	You have not showered in days	5	0	2

FOREIGN KEYS

WHAT IS A FOREIGN KEY?

- A foreign key (FK) is a column within a relational database table that provides a link betw
- A foreign key references
- Most relational database
tables together to prese

o link



FOREIGN KEYS

```
SELECT * FROM todos;
```

Select ALL columns and rows

Id (PK)	title	description	priority	complete	owner (FK)
1	Go to store	To pick up eggs	4	0	1
2	Haircut	Need to get length 1mm	3	0	1
3	Feed dog	Make sure to use new food brand	5	0	2
4	Water plant	Inside and Outside plants	4	0	2
5	Learn something new	Learn to program	5	0	1
6	Shower	You have not showered in days	5	0	2

FOREIGN KEYS

```
SELECT * FROM users;
```

Select ALL columns and rows

Id (PK)	email	username	first_name	last_name	hashed_password	is_active
1	codingwithroby@gmail.com	codingwithroby	Eric	Roby	123abcEqw!..	1
2	exampleuser12@example.com	exampleuser	Example	User	Gjjjd!2!..	1

- Each API request a user will have their ID attached
  - If we have the user ID attached to each request, we can use the ID to find their todos

Id (PK)	email	username	first_name	last_name	hashed_password	is_active
1	codingwithroby@gmail.com	codingwithroby	Eric	Roby	123abcEqw!..	1
2	exampleuser12@example.com	exampleuser	Example	User	Gjjjd!2!..	1

Id (PK)	title	description	priority	complete	owner (FK)
1	Go to store	To pick up eggs	4	0	1
2	Haircut	Need to get length 1mm	3	0	1
3	Feed dog	Make sure to use new food brand	5	0	2
4	Water plant	Inside and Outside plants	4	0	2
5	Learn something new	Learn to program	5	0	1
6	Shower	You have not showered in days	5	0	2

WHERE SQL QUERIES

```
SELECT * FROM todos WHERE owner=1;
```

Select ALL rows & columns WHERE owner = 1

Id (PK)	title	description	priority	complete	owner
1	Go to store	To pick up eggs	4	0	1
2	Haircut	Need to get length 1mm	3	0	1
5	Learn something new	Learn to program	5	0	1

WHERE SQL QUERIES

```
SELECT * FROM todos WHERE owner=2;
```

Select ALL rows & columns WHERE owner = 1

Id (PK)	title	description	priority	complete	owner
3	Feed dog	Make sure to use new food brand	5	0	2
4	Water plant	Inside and Outside plants	4	0	2
6	Shower	You have not showered in days	5	0	2

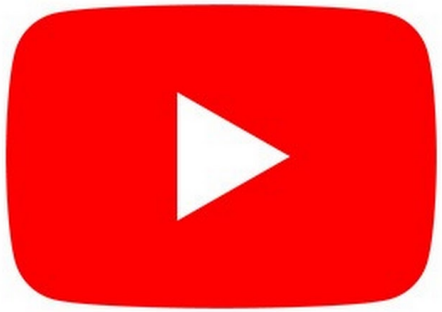
MYSQL INTRODUCTION

WHAT IS MYSQL

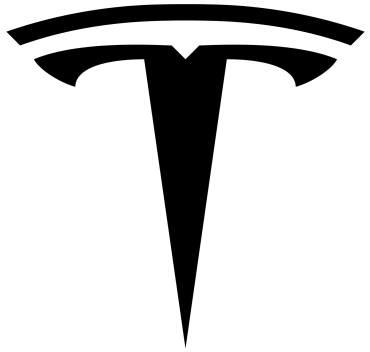
- Open-source relational database management system
- Requires a server
- Production ready
- Scalable
- Secure

WHO USES/USED MYSQL?

Facebook



YouTube

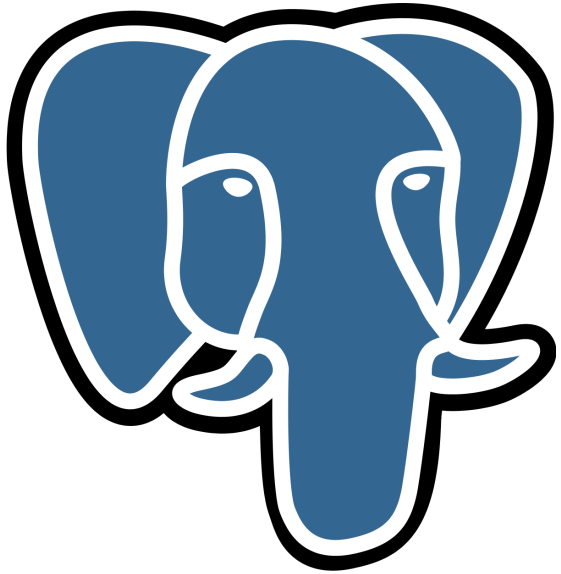


Tesla

POSTGRESQL INTRODUCTION

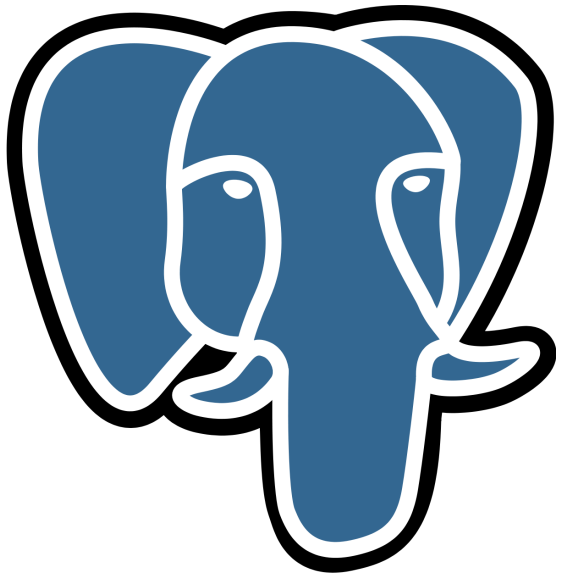
WHAT IS POSTGRESQL

- Production ready
- Open-source relational database management system
- Secure
- Requires a server
- Scalable



WHAT WILL WE COVER?

- How to install PostgreSQL
  - Windows
  - Mac
- Setup SQL tables
- Connect PostgreSQL to our application



WHO USES/USED POSTGRESQL?

Appl
e

Reddit

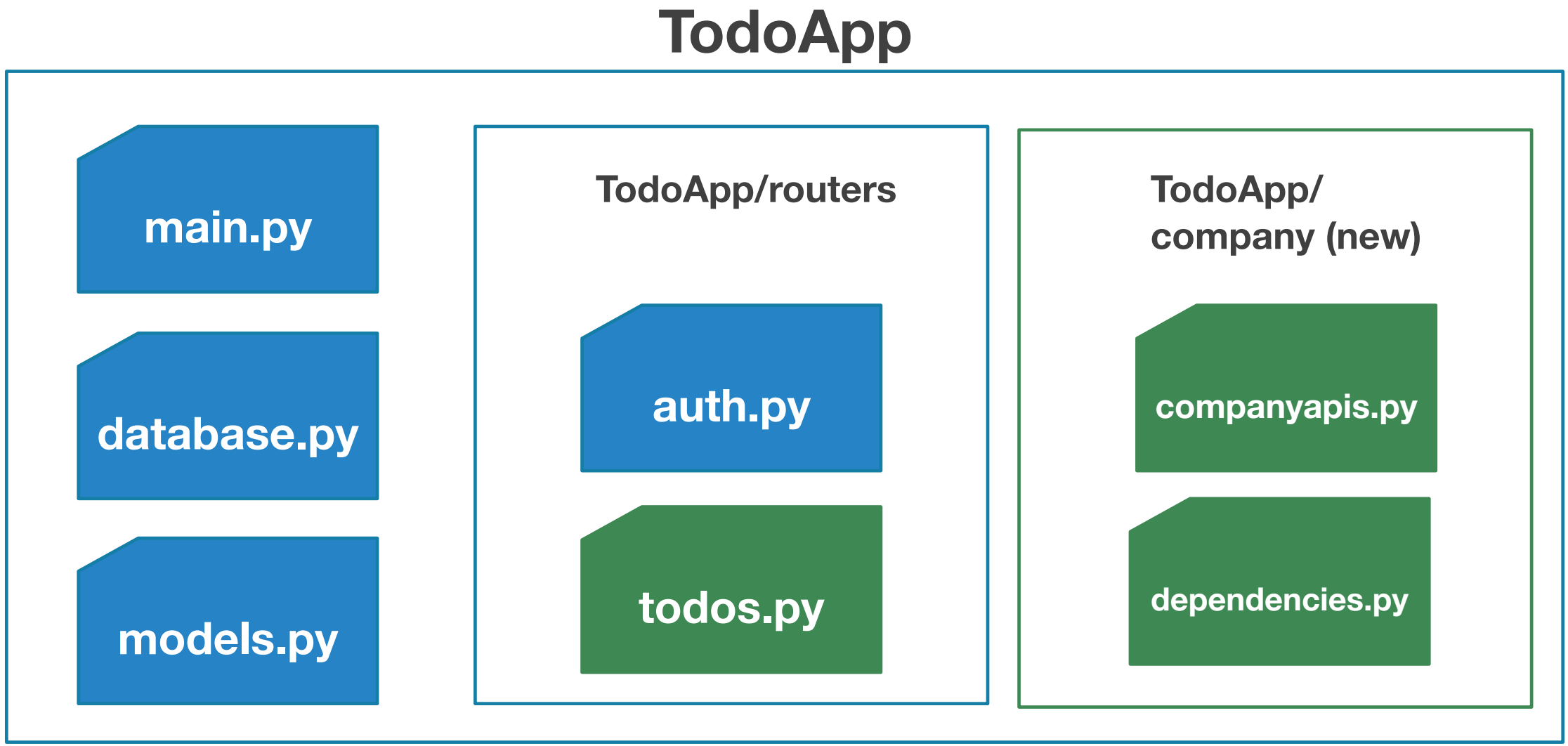
Twitch

ROUTING INTRODUCTION

WHAT IS ROUTING

- Rare that you want your entire application to be on a single file
- Flexible tool to structure your application
- Scalable architecture
- Organize file structure

NEW PROJECT STRUCTURE



auth

POST /auth/create/user Create New User

POST /auth/token Login For Access Token

todos

GET /todos/ Read All

POST /todos/ Create Todo

GET /todos/user Read All By User

GET /todos/{todo_id} Read Todo

PUT /todos/{todo_id} Update Todo

DELETE /todos/{todo_id} Delete Todo

companyapis

GET /companyapis/ Get Company Name

GET /companyapis/employees Number Of Employees

JINJA SCRIPTS

WHAT JINJA?

- Fast, expressive and extensible templating language
- Able to write code similar to Python in the DOM
- The template is passed data to render within the final document

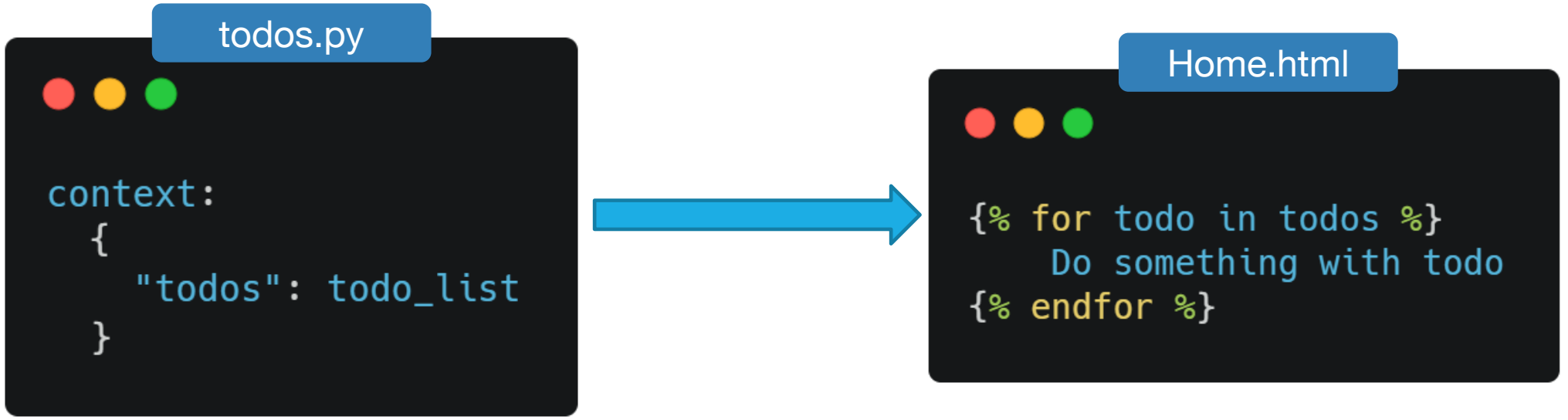
WHAT ARE JINJA TEMPLATING TAGS AND SCRIPTS?

- Jinja tags allows developers to be confident while working with backend data, using tags that are similar to HTML.

```
<link rel="stylesheet" type="text/css" href="{{ url_for('static', path='/todo/css/base.css') }}">
```

WHAT ARE JINJA TEMPLATING TAGS AND SCRIPTS?

- Now image we have a list of *todos* that we retrieved from the database. We can pass the entire list of *todos* into the front-end and loop through each *todo* with this simple ‘for loop’ on the template.



WHAT ARE JINJA TEMPLATING TAGS AND SCRIPTS?

- We can also use Jinja templating language with *if else* statements. One thing that may stand out is the double brackets with *todos|length*

```
{% if todos %}
    Displaying: {{ todos|length }} Todos
{% else %}
    You don't have any todos :)
{% endif %}
```

LET’S LEARN GIT

WHAT IS GIT?

- Free and open-source distributed **version control system**
- Can handle small to large applications and projects
- Allows team members to use same files by distributed branches/ environments

GIT EXAMPLE?

Version 1

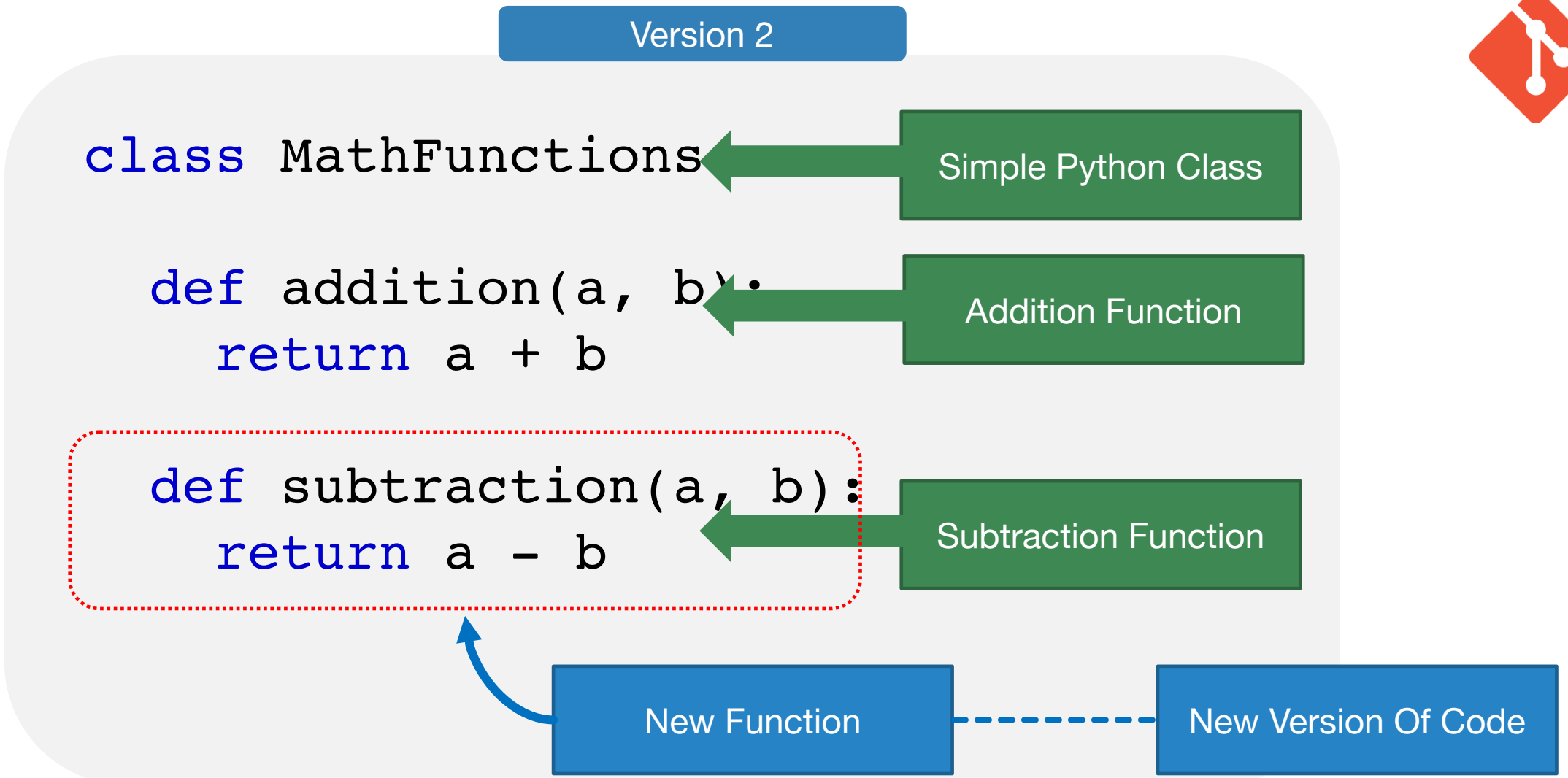
```
class MathFunctions

  def addition(a, b):
    return a + b
```

Simple Python Class

Addition Function

We need to add subtraction to MathFunctions

GIT EXAMPLE?



GIT EXAMPLE?

Version 3

```
class MathFunctions

  def addition(a, b):
    return a + b

  def subtraction(a, b):
    return a - b

  def multiplication(a, b):
    return a * b
```

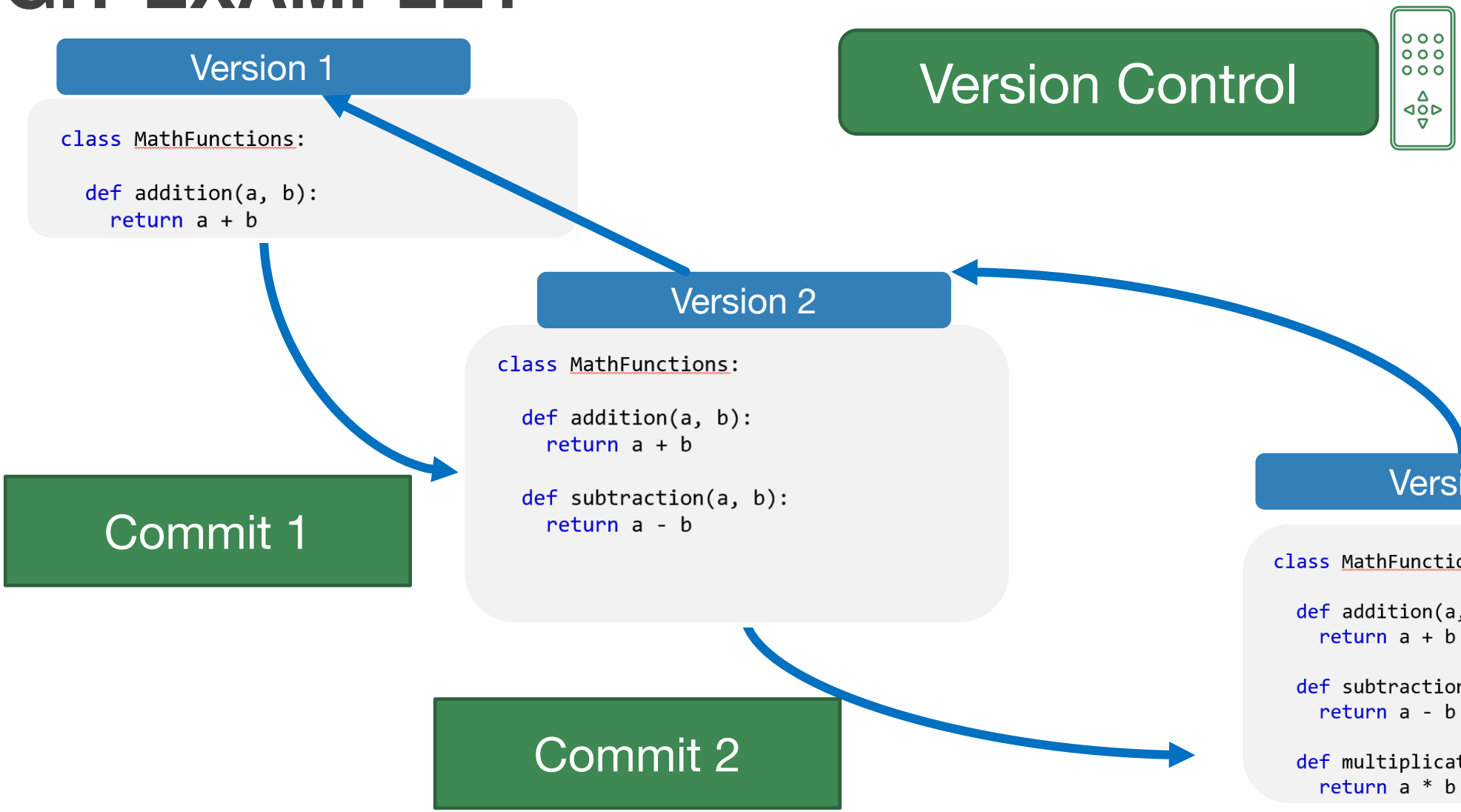
Simple Python Class

Addition Function

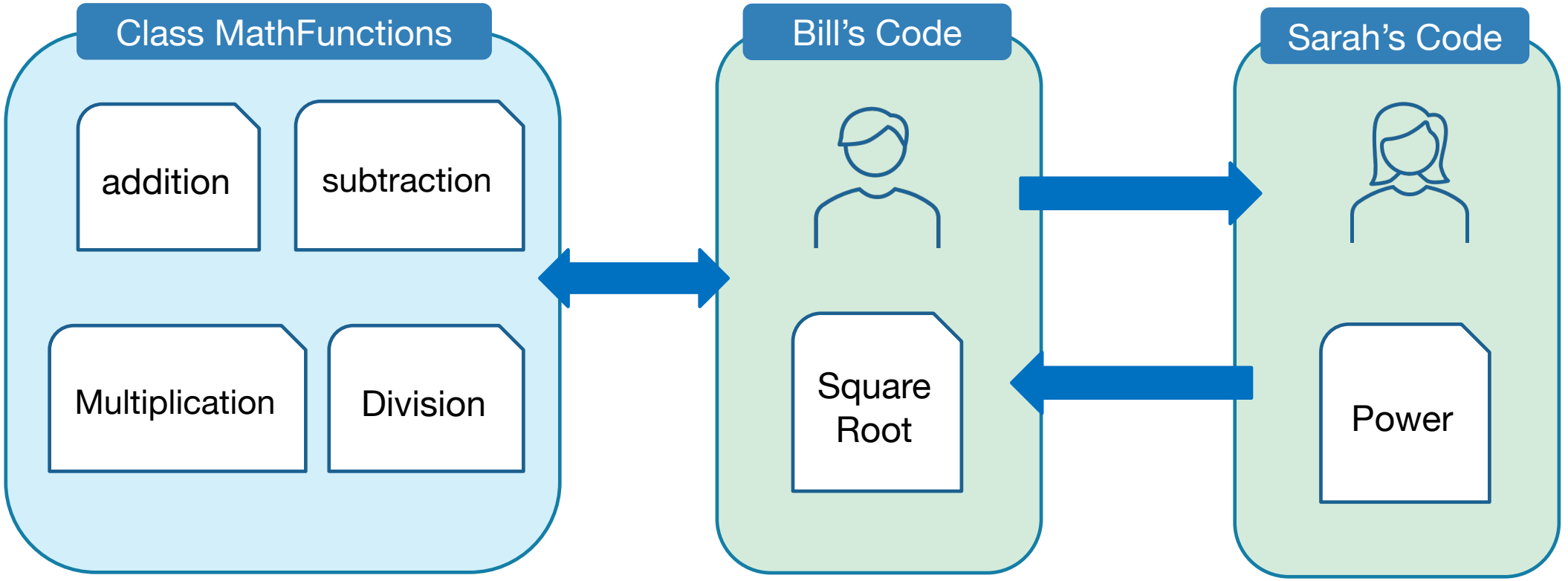
Subtraction Function

Multiple Function

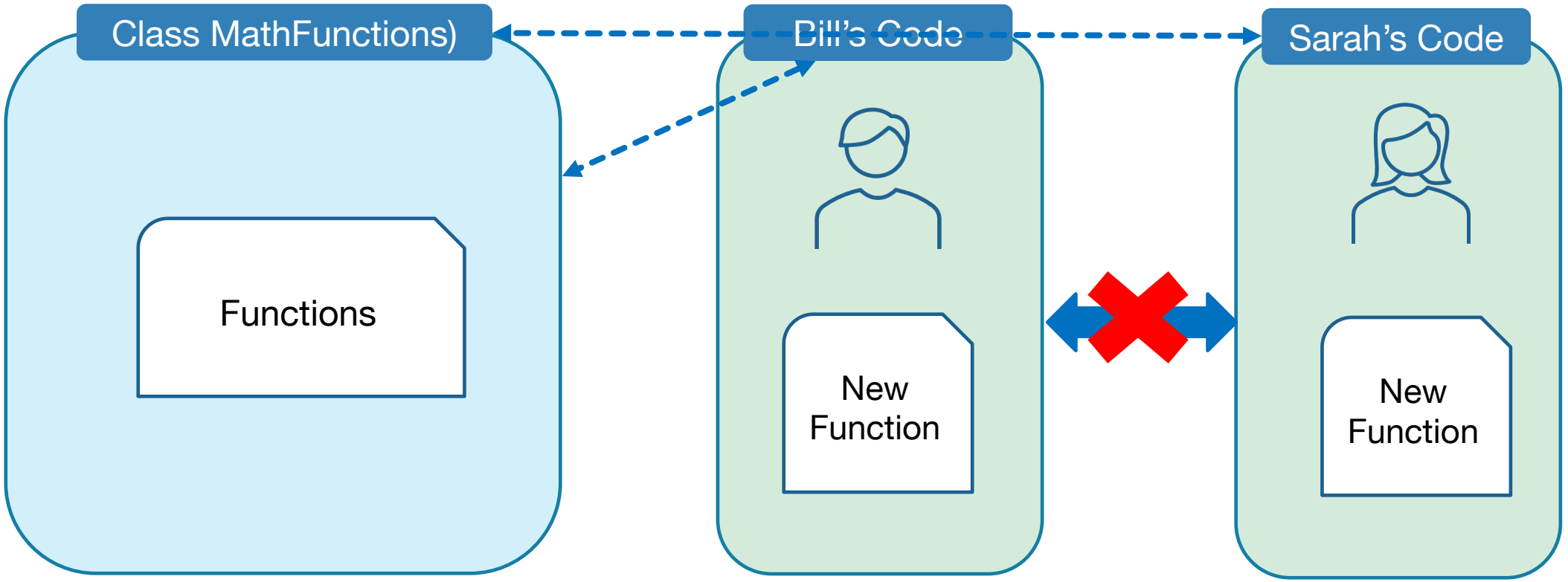
GIT EXAMPLE?



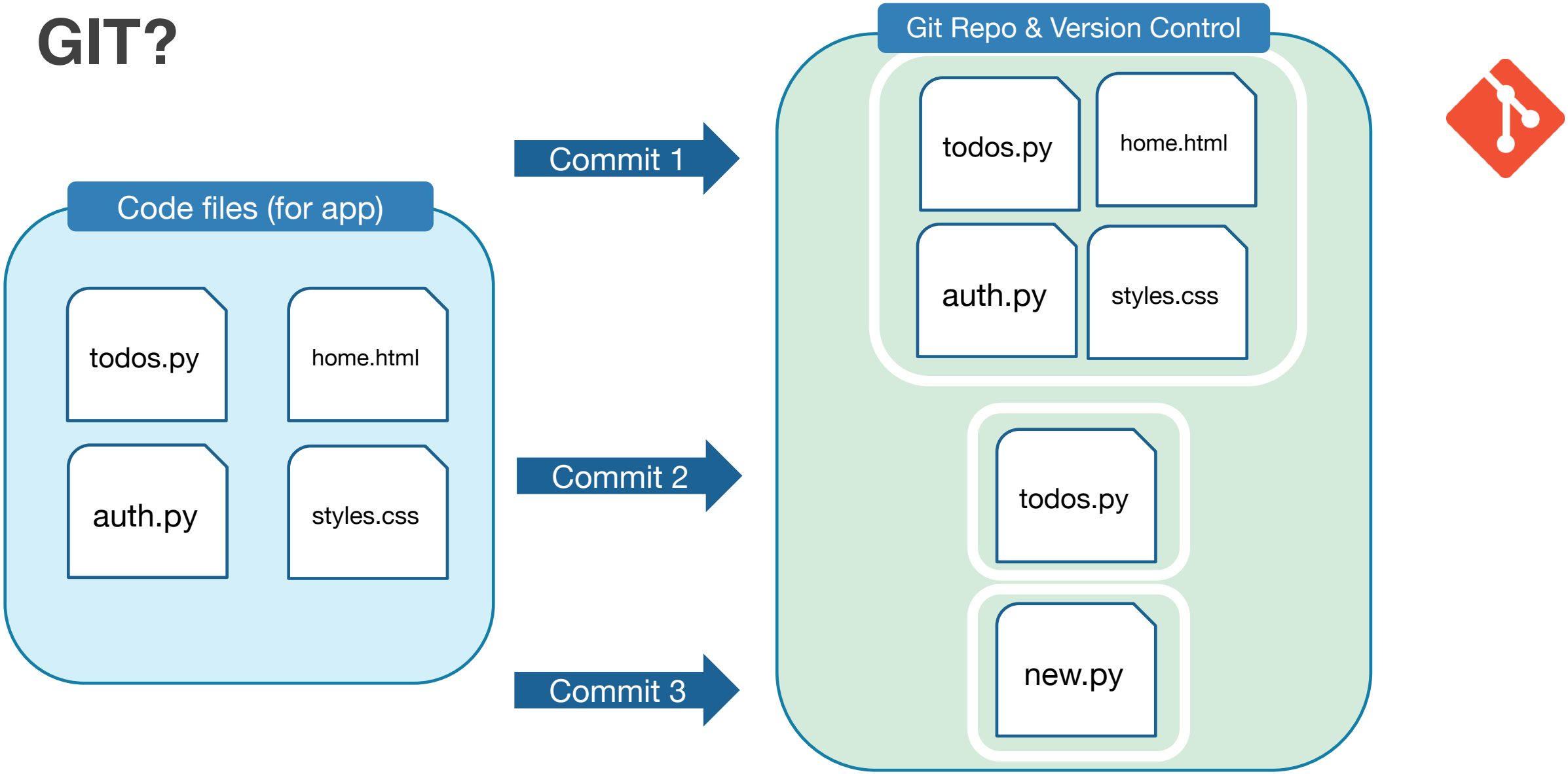
WHAT IS GIT?



WHAT IS GIT?

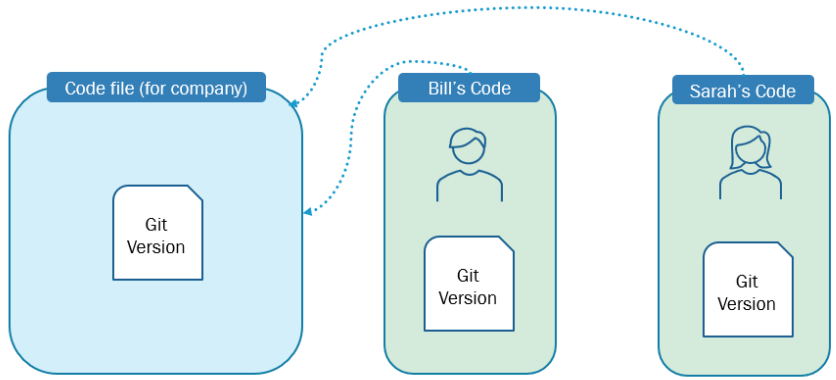


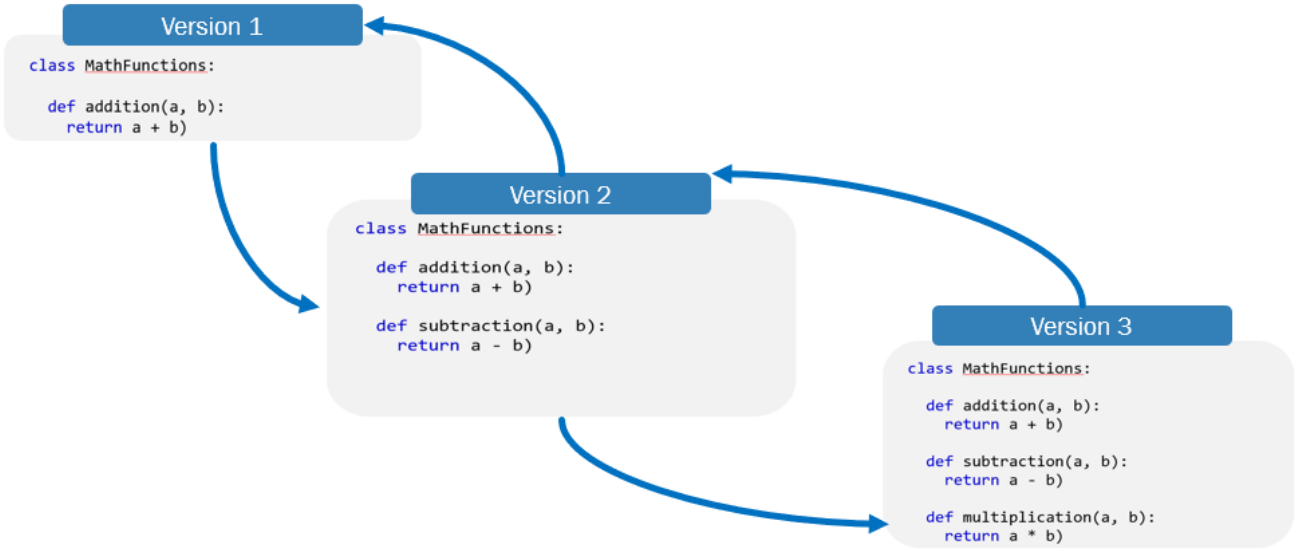
GIT?



WHAT IS GIT?

Free and open-source distributed version control system





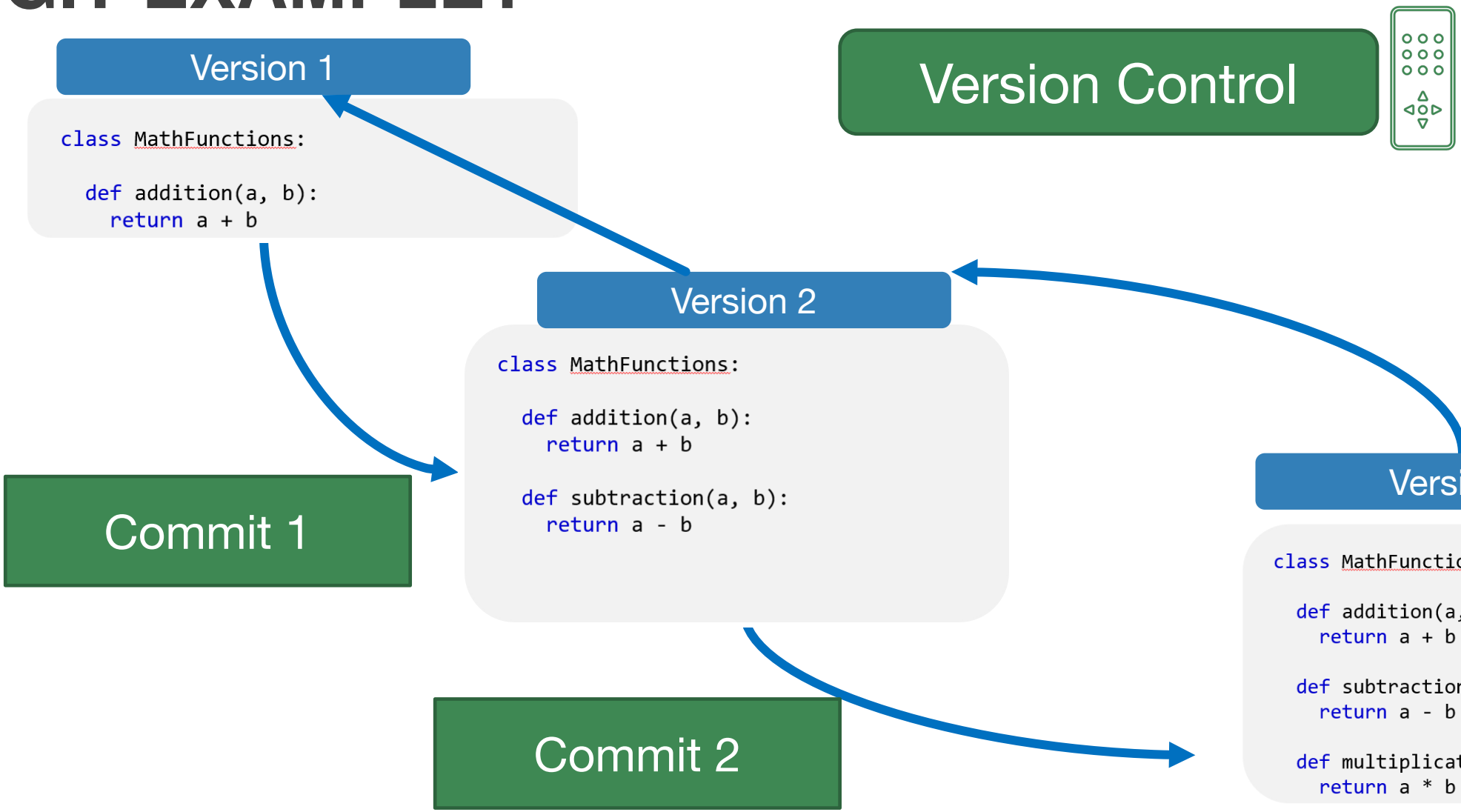
WHAT IS GIT?

Free and open-source distributed version control system

- Track Changes
- Version Control
- Allows team members to use same files without needing to sync every time

GIT BASICS

GIT EXAMPLE?



GIT EXAMPLE?

Git Command	Details
git init	Initializes a new, empty repository

GIT EXAMPLE?

Git Command	Details
git init	Initializes a new, empty repository
git add .	Adds files from a non-staged area to a staging GIT area

GIT EXAMPLE?

Git Command	Details
git init	Initializes a new, empty repository
git add .	Adds files from a non-staged area to a staging GIT area
git commit –m “info here”	Move files from a staging area to a commit

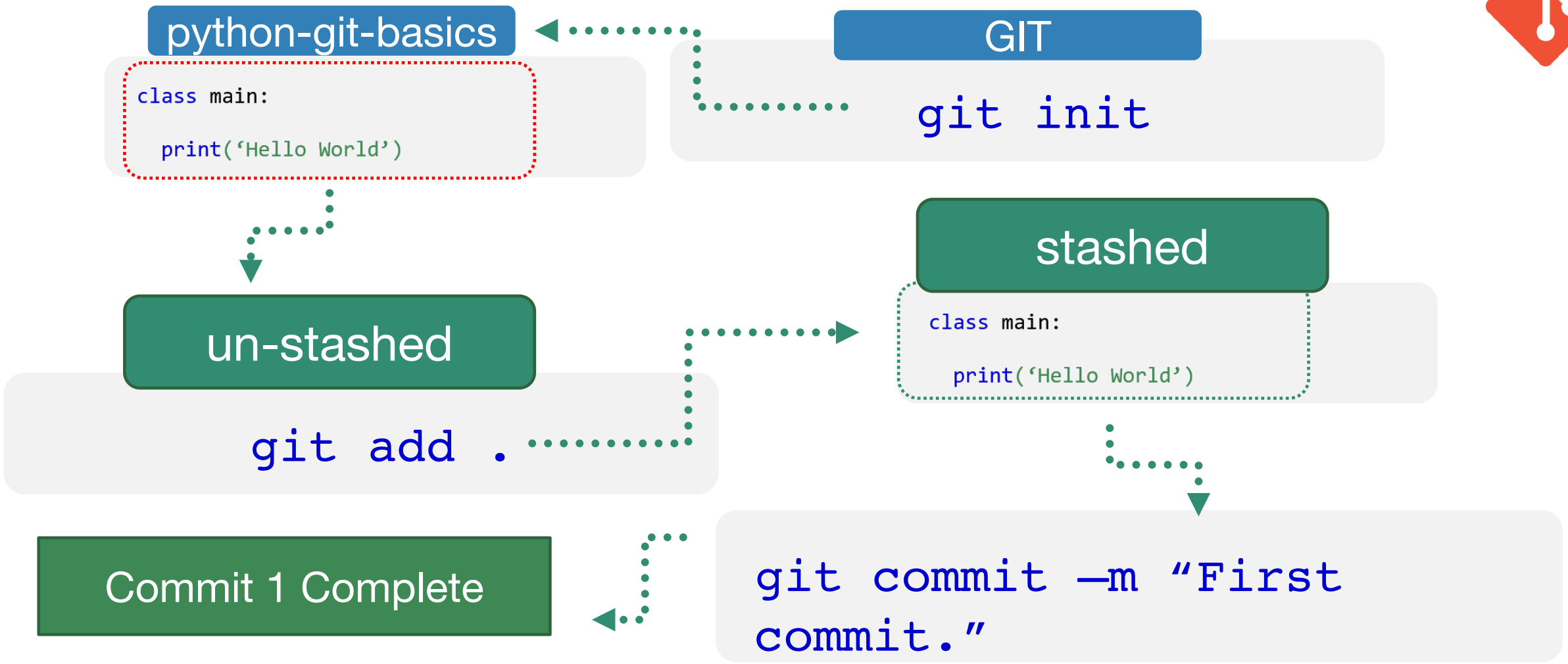
GIT EXAMPLE?

Git Command	Details
git init	Initializes a new, empty repository
git add .	Adds files from a non-staged area to a staging GIT area
git commit –m “info here”	Move files from a staging area to a commit
git checkout <commit #>	Open up previous commit

GIT EXAMPLE?

Git Command	Details
git init	Initializes a new, empty repository
git add .	Adds files from a non-staged area to a staging GIT area
git commit –m “info here”	Move files from a staging area to a commit
git checkout <commit #>	Open up previous commit
git log	Shows committed Snapshots

GIT BASICS



GIT BASICS

python-git-basics

```
class main:

  print('Hello World')

  print(2 + 2)
```

stashed

```
class main:

  print('Hello World')

  print(2 + 2)
```

un-stashed

```
git add .
```

```
git commit –m "Added addition functionality"
```

Commit 2 Complete

GIT BASICS

2 Different Commits

python-git-basics

```
git log
```

```
commit
aabbccddee112233445566
Author: Eric Roby
“Added Addition Function”

commit
aabbccddee112233445567
Author: Eric Roby
“First Commit”
```

```
git checkout
aabbccddee112233445567
```

GIT BRANCHES

WHAT IS A GIT BRANCH?

A pointer - to take a snapshot of a change. Branches can be merged into other branches

- A pointer of data change
- Isolation of feature development
- Linear development

GIT EXAMPLE?

Git Command	Details
git branch <branch name>	Create new isolated branch

GIT EXAMPLE?

Git Command	Details
git branch <branch name>	Create new isolated branch
git checkout branch <branch name>	Change root to branch selected

GIT EXAMPLE?

Git Command	Details
git branch <branch name>	Create new isolated branch
git checkout branch <branch name>	Change root to branch selected
git switch branch <branch name>	Same as above new command as of Git (2.23)

GIT BRANCHES?

All code is currently in the ***master branch***

```
git branch multiplication-branch
```

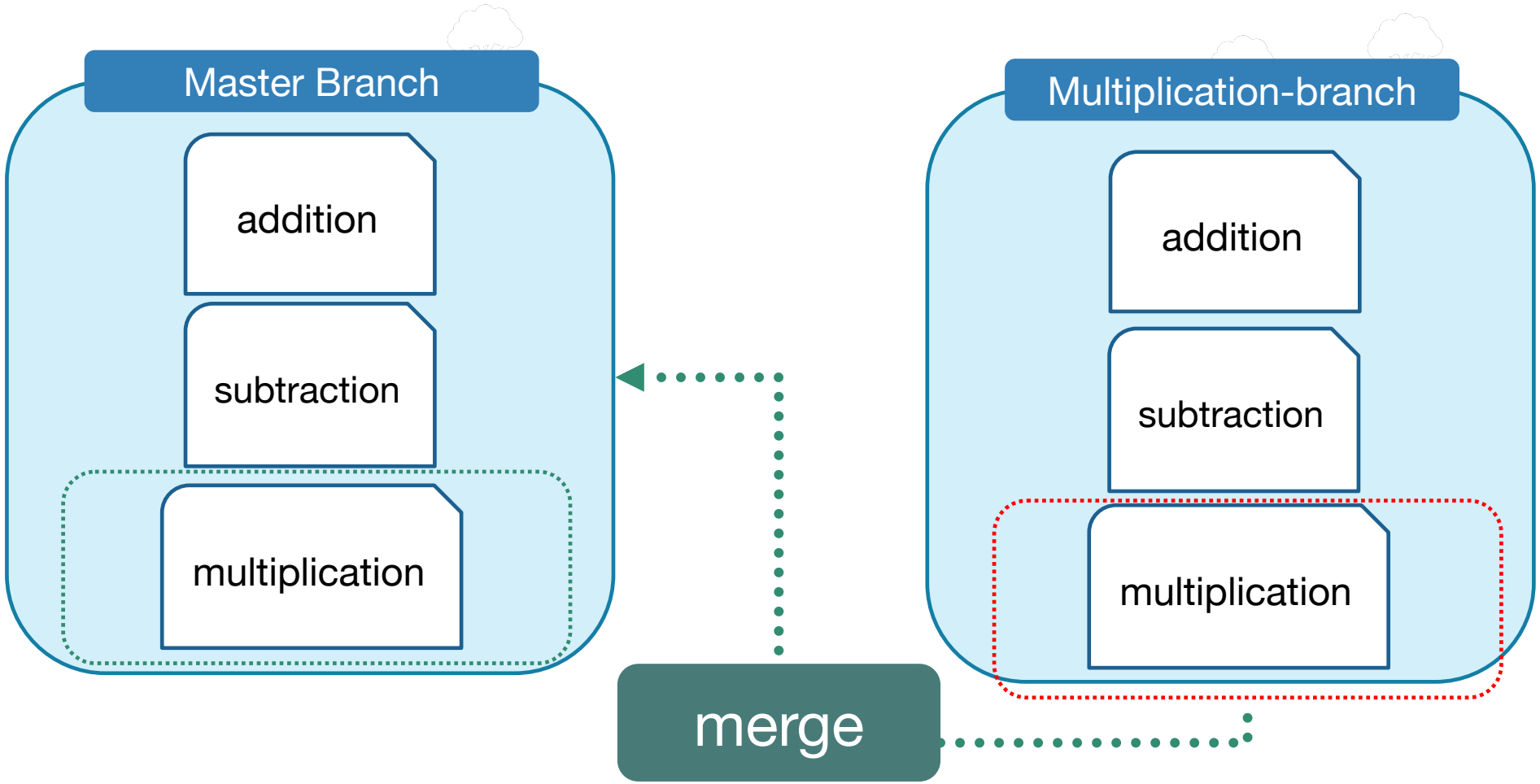
New branch

```
git checkout multiplication-branch
```

Checkout new branch

master branch ← → Multiplication-branch

GIT EXAMPLE?



GIT BRANCHES

python-git-basics

```
class MathFunctions:

  def addition(a, b):
    return a + b

  def subtraction(a, b):
    return a - b
```

detached-head

merge

master branch

```
git merge <commit id>
```

git checkout multiplication-branch

multiplication-branch

```
class MathFunctions:

  def addition(a, b):
    return a + b

  def subtraction(a, b):
    return a - b

  def multiplication(a, b):
    return a * b
```

```
git commit –m “adding multiplication function.”
```

master branch

```
git merge multiplication-branch
```

WHAT IS GITHUB?

WHAT IS GITHUB?

- Git Repository hosting service
- User friendly interface
- Large development platform

GitHub

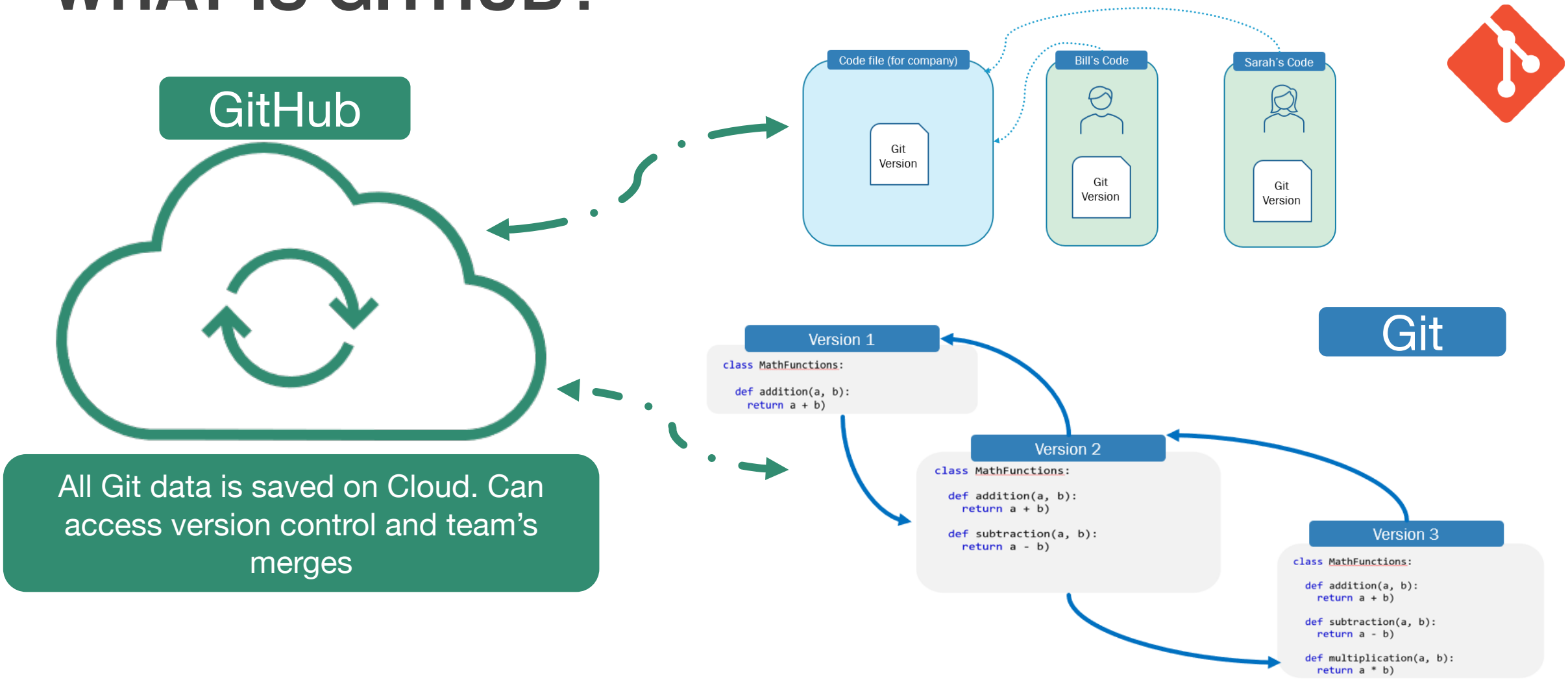
WHAT IS GITHUB?

git

Free and open-source distributed version control system

Top Git Repository hosting service available

WHAT IS GITHUB?



All Git data is saved on Cloud. Can access version control and team's merges

WHAT IS RENDER?

WHAT IS RENDER?

render

- Platform as a service (PaaS)
- Helps developers build, run and operate applications entirely on the cloud
- Developers can focus on coding and not have to worry about the infrastructure of their applications

RENDER PRICING?

render

- Pricing depends on many factors of how you want your application to perform online
- Free Trial
- Free Plan
- Businesses of all sizes use Render as their cloud PaaS provider.

WHAT IS RENDER?

render

- Code deployment system
- Continuous Integration & Continuous Deployment (CI/CD)
- Load Balancing
- and more…

Focus on Code

WHAT IS RENDER?

render

Version 1

```
class MathFunctions:

  def addition(a, b):
    return a + b)
```

Version 2

```
class MathFunctions:

  def addition(a, b):
    return a + b)

  def subtraction(a, b):
    return a - b)
```

Version 3

```
class MathFunctions:

  def addition(a, b):
    return a + b)

  def subtraction(a, b):
    return a - b)

  def multiplication(a, b):
    return a * b)
```

render

GitHub

DIFFERENT PLATFORMS?

render

THANK YOU!!

CONTACT ME!

- codingwithroby@gmail.com
- Instagram: @codingwithroby

LET'S LEARN PYTHON

Assignment!

LET'S LEARN PYTHON

Variables!

LET'S LEARN PYTHON

Comments!

LET'S LEARN PYTHON

String

LET'S LEARN PYTHON

User Input!

LET'S LEARN PYTHON

Lists!

LET'S LEARN PYTHON

Sets & Tuples!

LET'S LEARN PYTHON

Boolean &

LET'S LEARN PYTHON

If Else!

LET'S LEARN PYTHON

Loops!

LET'S LEARN PYTHON

Dictionaries!

LET'S LEARN PYTHON

Functions!