



COM5940: NEW MEDIA BUSINESS MODEL & INNOVATION INTEGRATE JS FRONT-END WITH FLASK REST API BACK-END SERVICES

Bernard Suen

Center for Entrepreneurship

Chinese University of Hong Kong



Center for
Entrepreneurship

Today's Agenda

1. Use **FlaskAlchemy** for MySQL data connection and CRUD support.
2. Use **Airtable** APIs for providing CRUD and **authentication** support.
3. Utilize JSON data in **jQuery Datatable**, **Leaflet** & **C3**.
4. Enable **RESTful API** by turning MySQL data into JSON for connection with mobile front-end.
5. **Unit, integration** and **user acceptance** tests.

**Introducing
Flask
SQLAlchemy**

**CRUD Support in
SQLAlchemy and
Airtable for
View Integration**

**Unit Testing
Integration Testing
User Acceptance
Testing**

**Why FlaskSQLAlchemy if we already have
MySQLdb or Pymysql?**

1. Automatic **SQL mapping** to Python objects, so no need to know SQL. But one has to learn the object naming convention.
2. Database independent so once one learns how to use it on MySQL, it will work almost the same on Oracle, MSSQL, PostgreSQL, etc.
3. Practical and simple **pagination** support.
4. The reason why **FlaskSQLAlchemy** wasn't taught first instead of Pymysql and MySQLdb is because I think it's important for you to learn SQL.



Adobe Creative
Cloud for Teams
starting at \$33.99
per month.

ads via Carbon

Dialects

The **dialect** is the system SQLAlchemy uses to communicate with various types of **DBAPI** implementations and databases. The sections that follow contain reference documentation and notes specific to the usage of each backend, as well as notes for the various DBAPIs.

All dialects require that an appropriate DBAPI driver is installed.

Included Dialects

- PostgreSQL
- MySQL
- SQLite
- Oracle
- Microsoft SQL Server

<https://docs.sqlalchemy.org/en/13/dialects/>

Included, but not currently supported dialects

The following dialects have implementations within SQLAlchemy, but they are not part of continuous integration testing nor are they actively developed. These dialects may be removed in future major releases.

[Firebird](#)

<https://docs.sqlalchemy.org/en/13/dialects/>

Utilities

`class flask_sqlalchemy.Pagination(query, page, per_page, total, items)`

Internal helper class returned by `BaseQuery.paginate()`. You can also construct it from any other SQLAlchemy query object if you are working with other libraries. Additionally it is possible to pass `None` as query object in which case the `prev()` and `next()` will no longer work.

`has_next`

True if a next page exists.

`has_prev`

True if a previous page exists

`items = None`

the items for the current page

`iter_pages(left_edge=2, left_current=2, right_current=5, right_edge=2)`

Iterates over the page numbers in the pagination. The four parameters control the thresholds how many numbers should be produced from the sides. Skipped page numbers are represented as `None`. This is how you could render such a pagination in the templates:

<https://flask-sqlalchemy.palletsprojects.com/en/2.x/api/>



Quit

Logout

Duplicate | Rename | Move | Download | View | Edit |

Upload

New ▾



flask_apps / RESTful Flask Apps / Clear Blog Flask Demo / Login Authentication 2

Name

Last Modified

File size

..		seconds ago
How-to-Get-Started-with-Scraping-in-Python		4 days ago
projects		2 years ago
static		3 hours ago
templates		2 hours ago
Flask-HttpAuth_Test1.ipynb	7 days ago	2.13 kB
Python Data Science Learning Notebook (Matplotlib, Pandas, Numpy, and SciPy .ipynb)	4 days ago	550 kB
Scrape, Transform, and Store.ipynb	3 days ago	217 kB
scrapy_test.ipynb	2 days ago	25.6 kB
test-basic auth-1.ipynb	3 days ago	9.46 kB
test-basic auth-2.ipynb	5 hours ago	9.61 kB
test-basic auth-3.ipynb	Running 5 minutes ago	12.5 kB
test_flasksqlalchemy.ipynb	Running 8 hours ago	12 kB
test_jwt1.ipynb	4 hours ago	10.3 kB
kenneth2.csv	3 days ago	10.2 kB
<input checked="" type="checkbox"/> product_model.py	a day ago	618 B
quotejl.pickle	6 days ago	5.48 kB
quoteresult.jl	6 days ago	5.55 kB
quoteresult.json	6 days ago	5.57 kB
startup_beat_data_1.csv	3 days ago	10.2 kB

product_model.py

```
from flask import Flask
from flask_sqlalchemy import SQLAlchemy

app = Flask(__name__)

app.config['SQLALCHEMY_DATABASE_URI'] = 'mysql://root:root@localhost/southwind'
app.config['SECRET_KEY'] = "mysecret"
app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = False
db = SQLAlchemy(app)

class Products(db.Model):
    __tablename__ = 'products'
    productID = db.Column(db.Integer, primary_key=True)
    productCode = db.Column(db.String(3))
    name = db.Column(db.String(30))
    quantity = db.Column(db.Integer)
    price = db.Column(db.Float)
    supplierID = db.Column(db.Integer)
    # description = db.Column(db.Text)
```

Similar to how we setup MySQL database using py mysql and MySQLdb, we use the following codes to set up Flask SQLAlchemy to connect to the MySQL database.

Since SQLAlchemy is a object-based (i.e. class) database mapper, it uses object notation to carry out its operations.

```
1 from product_model import * ←  
  
1 result = db.engine.execute('select * from products')  
2 for i in result:  
3     print(i)  
4 result = db.engine.execute('select * from products')  
5 final_result = [list(i) for i in result]  
6 dataset = []  
7 dict = {}  
8 for i in final_result:  
9     dict['productID'] = i[0]  
10    dict['productCode'] = i[1]  
11    dict['name'] = i[2]  
12    dict['quantity'] = i[3]  
13    dict['price'] = i[4]  
14    dict['supplierID'] = i[5]  
15    # print(i)  
16    # print(dict)  
17    dataset.append(dict.copy()) #markers.append(fld.copy())  
18 print(dataset)
```

Include the `product_model.py` module in this program.

But we can still use SQL if we want.
SQLAlchemy does support SQL statements.

```
1 # Update Example  
2 pid = 1003  
3 products = Products.query.filter_by(productID=pid).first()  
4 products.productCode = "SAT"  
5 db.session.commit()
```

```
1 # Insert Example  
2 product = Products(productCode='EGG', name="Bernard", quantity=100, price=1.5, supplierID=502)  
3 db.session.add(product)  
4 db.session.commit()
```

```
1 # Delete Example  
2 pid = 2044  
3 product = Products.query.filter_by(productID=pid).first()  
4 db.session.delete(product)  
5 db.session.commit()
```

In order to support paging, we need to ensure that our database has a primary key and support automatic increment of key ID.

[Recent](#)[Favorites](#)

Filter databases by name X

- + service-leadership
- + southwind
 - New
- + coronavirus
- products**
- + suppliers
- + table1
- + suenlab1_6d8
- + test
- + users
- + webzash
- + willey
- + wp52a
- + wp475_c
- + wp498a
- + wp498b
- + wp498c
- + wp510b
- + wp510c
- + wp510d
- + wp510e

Server: localhost:3306 » Database: southwind » Table: products

Browse Structure SQL Search Insert Export Import Privileges Operations Triggers

Table structure Relation view

#	Name	Type	Collation	Attributes	Null	Default	Extra	Action
1	productID	int(4)			Yes	NULL		Primary Unique More
2	productCode	varchar(3)	utf8_general_ci		Yes	NULL		Primary Unique More
3	name	varchar(9)	utf8_general_ci		Yes	NULL		Primary Unique More
4	quantity	int(5)			Yes	NULL		Primary Unique More
5	price	decimal(7,2)			Yes	NULL		Primary Unique More
6	supplierID	int(3)			Yes	NULL		Primary Unique More

 Check All

With selected:

[Browse](#) [Change](#) [Drop](#) [Primary](#) [Unique](#) [Index](#)[Print view](#)[Propose table structure](#) [Move columns](#) [Improve table structure](#) [Add 1](#)

column(s) after supplierID

[Go](#)[+ Indexes](#)

Information

Space usage

Row statistics

Data	16	KiB	Format	Compact
Index	0	B	Collation	utf8_general_ci
Total	16	KiB	Creation	Mar 06, 2020 at 08:54 PM

[Console](#)

Recent Favorites
Filter databases by name X

- service-leadership
- southwind
 - New
- coronavirus
- products**
- suppliers
- table1
- suenlab1_6d8
- test
- users
- webzash
- willey
- wp52a
- wp475_c
- wp498a
- wp498b
- wp498c
- wp510b
- wp510c
- wp510d
- wp510e

Server: localhost:3306 » Database: southwind » Table: products

Browse Structure SQL Search Insert Export Import Privileges Operations Triggers

+ Options

productID	productCode	name	quantity	price	supplierID
1001	PEN	Pen Red	5000	1.23	501
1002	PEN	Pen Blue	8000	1.25	501
1003	PEN	Pen Black	2000	1.25	501
1004	PEC	Pencil 2B	10000	0.48	501
1005	PEC	Pencil 2H	8000	0.49	502
1006	PEC	Pencil HB	0	99999.99	501
1007	PEN	Pen Red	5000	1.23	501
1008	PEN	Pen Blue	8000	1.25	501
1009	PEN	Pen Black	2000	1.25	501
1010	PEC	Pencil 2B	10000	0.48	501
1011	PEC	Pencil 2H	8000	0.49	502
1012	PEC	Pencil HB	0	99999.99	501
1013	PEN	Pen Red	5000	1.23	501
1015	PEN	Green 2B	20	2.50	502
1016	PEC	Green 2B	43	3.50	501
2001	PEC	Pencil 3B	500	0.52	503
2002	PEC	Pencil 4B	200	0.62	501
2003	PEN	Pencil 5B	100	0.73	503
2004	PEC	Pencil 6B	500	0.47	502
2005	PEC	Pencil 3B	500	0.52	503
2006	PEC	Pencil 4B	200	0.62	501
2007	PEN	Pencil 5B	100	0.73	503
2008	PEN	Pencil 6B	500	0.47	502
2009	PEC	Pencil 3B	500	0.52	503
2010	PEC	Pencil 4B	200	0.62	501
2011	PEC	Pencil 5B	100	0.73	503
2012	PEC	Pencil 6B	500	0.47	502
2009	PEC	Pencil 3B	500	0.52	503
2010	PEC	Pencil 4B	200	0.62	501
Console 1	PEC	Pencil 5B	100	0.73	503

Paging through the Database Using Built-in Parameters

```

1  {% extends "base.html" %} 
2  {% block content %} 
3 
4  {% for product in products.items %} 
5      {% if loop.index == 1 %}{% set counter = ( counter | default(1) ) %}{% endif %} 
6      {% if counter == 1 %}<div class="row">{% endif %} 
7          <div class="col-md-4"> 
8              <div class="thumbnail"> 
9                  <p>Col#:{{ loop.index }}<br>Product Code:{{ product.productID }}<br>Quantity:{{ product.quantity }}<br>Price:{{ 
product.price }}</p> 
10                 <div class="btn-group"> 
11                     <button type="button" class="btn btn-sm btn-outline-secondary">View</button> 
12                 </div> 
13             </div> 
14         </div> 
15         {% if counter == 3 %}</div><!--end row-->{% set counter = ( counter | default(1) ) %}{% else %}{% set counter = ( counter | 
default(0) ) + 1 %}{% endif %} 
16     {% endfor %} 
17     <div class="container" style="float: left;"> 
18         {% for page in products.iter_pages() %} 
19             {% if page %} 
20                 <a href="{{url_for('album')}}/{{ page }}">{{ page }}</a> 
21             {% else %} 
22                 ... 
23             {% endif %} 
24         {% endfor %} 
25     </div> 
26 {% endblock %}

```

The second loop controls the paging of the album.

Two loops are involved here. The first loop goes through the rows and columns of the albums.

Col#:1
Product ID:1001
Product Code:PEN
Product NAME:Pen Red
Quantity:5000
Price:1.23
Supplier ID:501

[View](#)

Col#:2
Product ID:1002
Product Code:PEN
Product NAME:Pen Blue
Quantity:8000
Price:1.25
Supplier ID:503

[View](#)

Col#:3
Product ID:1003
Product Code:SAT
Product NAME:Pen God
Quantity:2000
Price:1.25
Supplier ID:501

[View](#)

Col#:4
Product ID:1004
Product Code:XYZ
Product NAME:Pen Pen
Quantity:10000
Price:9.75
Supplier ID:501

[View](#)

Col#:5
Product ID:1006
Product Code:PEC
Product NAME:Pencil HB
Quantity:222
Price:123.23
Supplier ID:501

[View](#)

Col#:6
Product ID:1007
Product Code:PEN
Product NAME:Pen Red
Quantity:5000
Price:1.23
Supplier ID:501

[View](#)

Col#:7
Product ID:1008
Product Code:PEN
Product NAME:Pen Blue
Quantity:8000
Price:1.25
Supplier ID:501

[View](#)

Col#:8
Product ID:1009
Product Code:PEN
Product NAME:Pen Black
Quantity:2000
Price:1.25
Supplier ID:501

[View](#)

Col#:9
Product ID:1010
Product Code:PEC
Product NAME:Pencil 2B
Quantity:10000
Price:0.48
Supplier ID:501

[View](#)

CRUD Operations in Action

Update Product Delete Product Add button						Search: <input type="text"/>
Product ID	Product Code	Name	Quantity	Price	Supplier ID	
1001	PEN	Pen Red	5000	1.23	501	
1002	PEN	Pen Blue	8000	1.25	503	
1003	SAT	Pen God	2000	1.25	501	
1004	XYZ	Pen Pen	10000	9.75	501	
1006	PEC	Pencil HB	222	123.23	501	
1007	PEN	Pen Red	5000	1.23	501	
1008	PEN	Pen Blue	8000	1.25	501	
1009	PEN	Pen Black	2000	1.25	501	
1010	PEC	Pencil 2B	10000	0.48	501	
1011	PEC	Pencil 2H	8000	0.49	502	

Showing 1 to 10 of 50 entries

[Previous](#) [1](#) [2](#) [3](#) [4](#) [5](#) [Next](#)

Update Product Delete Product Add button						<input type="text" value="Search:"/>					
Product ID	↓↑	Product Code	↓↑	Name	↓↑	Quantity	↓↑	Price	↓↑	Supplier ID	↓↑
1001		PEN		Pen Red		5000		1.23		501	
1002		PEN		Pen Blue		8000		1.25		503	
1003		SAT		Pen God		2000		1.25		501	
1004		XYZ		Pen Pen		10000		9.75		501	
1006		PEC		Pencil HB		222		123.23		501	
1007		PEN		Pen Red		5000		1.23		501	
1008		PEN		Pen Blue		8000		1.25		501	
1009		PEN		Pen Black		2000		1.25		501	
1010		PEC		Pencil 2B		10000		0.48		501	
1011		PEC		Pencil 2H		8000		0.49		502	

Showing 1 to 10 of 50 entries 1 row selected

[Previous](#) [1](#) [2](#) [3](#) [4](#) [5](#) [Next](#)

Update Product		Delete Product		Add button		Search: <input type="text"/>		
Product ID		Product Code		Name		Quantity	Price	Supplier ID
1001		PEN		Pen Red		5000	1.23	501
1002		PEN		Pen Blue		8000	1.25	503
1003		SAT		Pen God		2000	1.25	501
1004		XYZ		Pen Pen		10000	9.75	501
1006		PEC		Pencil HB		222	123.23	501
1007		PEN		Pen Red		5000	1.23	501
1008		PEN		Pen Blue		8000	1.25	501
1009		PEN		Pen Black		2000	1.25	501
1010		PEC		Pencil 2B		10000	0.48	501
1011		PEC		Pencil 2H		8000	0.49	502

Showing 1 to 10 of 50 entries

[Previous](#) [1](#) [2](#) [3](#) [4](#) [5](#) [Next](#)

Add Entry

Product Code

Name

Quantity

Price

Supplier ID

SubmitClose

Update Product		Delete Product	Add
Product ID	Product Name	Supplier ID	
1001	PEN		
1002	PEN		
1003	SAT		
1004	XYZ		
1006	PEC		
1007	PEN		
1008	PEN	Pen Blue	8000 1.25 501
1009	PEN	Pen Black	2000 1.25 501
1010	PEC	Pencil 2B	10000 0.48 501
1011	PEC	Pencil 2H	8000 0.49 502

Showing 1 to 10 of 50 entries

Search:

Supplier ID

Supplier ID
501
503
501
501
501
501
501
501
502

Previous12345Next

Update Product		Delete Product		Add button					Search: <input type="text"/>	
Product ID		Product Code		Name		Quantity		Price		Supplier ID
1001		PEN		Pen Red		5000		1.23		501
1002		PEN		Pen Blue		8000		1.25		503
1003		SAT		Pen God		2000		1.25		501
1004		XYZ		Pen Pen		10000		9.75		501
1006		PEC		Pencil HB		222		123.23		501
1007		PEN		Pen Red		5000		1.23		501
1008		PEN		Pen Blue		8000		1.25		501
1009		PEN		Pen Black		2000		1.25		501
1010		PEC		Pencil 2B		10000		0.48		501
1011		PEC		Pencil 2H		8000		0.49		502

Showing 1 to 10 of 50 entries 1 row selected

[Previous](#) [1](#) [2](#) [3](#) [4](#) [5](#) [Next](#)

Update Product		Delete Product	Add Product
Product ID	↓	Product Name	
1001		PEN	
1002		PEN	
1003		SAT	
1004		XYZ	
1006		PEC	
1007		PEN	
1008		PEN	
1009		PEN	
1010		PEC	
1011		PEC	

Update Entry

Product ID

1003

Product Code

SAT

Name

Pen God

Quantity

2000

Price

1.25

Supplier ID

501

SubmitClose

Showing 1 to 10 of 50 entries 1 row selected

Previous 1 2 3 4 5 Next

Update Product Delete Product Add button						<input type="text" value="Search:"/>
Product ID	Product Code	Name	Quantity	Price	Supplier ID	
1001	PEN	Pen Red	5000	1.23	501	
1002	PEN	Pen Blue	8000	1.25	503	
1003	SAT	Pen God	2000	1.25	501	
1004	XYZ	Pen Pen	10000	9.75	501	
1006	PEC	Pencil HB	222	123.23	501	
1007	PEN	Pen Red	5000	1.23	501	
1008	PEN	Pen Blue	8000	1.25	501	
1009	PEN	Pen Black	2000	1.25	501	
1010	PEC	Pencil 2B	10000	0.48	501	
1011	PEC	Pencil 2H	8000	0.49	502	

Showing 1 to 10 of 50 entries 1 row selected

[Previous](#) [1](#) [2](#) [3](#) [4](#) [5](#) [Next](#)

Delete Entry

Product ID

1003

Product Code

SAT

Name

Pen God

Quantity

2000

Price

1.25

Supplier ID

501

Submit



Close

Update Product		Delete Product		Add Product	
Product ID	Product Name	Product Code	Quantity	Supplier ID	Price
1001		PEN			
1002		PEN			
1003		SAT			
1004		XYZ			
1006		PEC			
1007		PEN			
1008		PEN			
1009		PEN			
1010		PEC			
1011		PEC			

Product ID	1003
Product Code	SAT
Name	Pen God
Quantity	2000
Price	1.25
Supplier ID	501

Search:

Supplier ID

Supplier ID	Product ID	Product Name	Quantity	Price	Category
501	1001	Pen Blue	5000	1.25	
503	1002	Pen Black	2000	1.25	
501	1003	Pencil 2B	10000	0.48	501
501	1004	Pencil 2H	8000	0.49	502
501	1006				
501	1007				
501	1008				
501	1009				
501	1010				
502	1011				

Showing 1 to 10 of 50 entries 1 row selected

[Previous](#) [1](#) [2](#) [3](#) [4](#) [5](#) [Next](#)

```

from flask import Flask, render_template, url_for, redirect, request, make_response, Response, jsonify
from flask_login import LoginManager, login_user, current_user, logout_user, login_required, UserMixin
from flask_cors import CORS
from flask_sqlalchemy import SQLAlchemy
from datetime import datetime
import requests
import jwt

##### Initialize Flask App #####
app = Flask(__name__)
CORS(app)

#### MySQL SQLAlchemy Object Relations Mapping #####
app.config['SQLALCHEMY_DATABASE_URI'] = 'mysql://root:root@localhost/southwind'
app.config['SECRET_KEY'] = "mysecret"
app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = False
db = SQLAlchemy(app)

class Products(db.Model):
    __tablename__ = 'products'
    productID = db.Column(db.Integer, primary_key=True)
    productCode = db.Column(db.String(3))
    name = db.Column(db.String(30))
    quantity = db.Column(db.Integer)
    price = db.Column(db.Float)
    supplierID = db.Column(db.Integer)
    # description = db.Column(db.Text)

#####
# Login Manager Setup #####
login_manager = LoginManager()
login_manager.init_app(app)
login_manager.login_view = 'login'
app.config['SECRET_KEY'] = "lkkajdghdadkglajkgah"

@login_manager.user_loader
def load_user(user_id):
    return User(user_id)

class User(UserMixin):
    def __init__(self,id):
        self.id = id

```

Similar to how we setup MySQL database using py mysql and MySQLdb, we use the following codes to set up Flask SQLAlchemy to connect to the MySQL database.

Since SQLAlchemy is a object-based (i.e. class) database mapper, it uses object notation to carry out its operations.

```
@app.route("/list_album")
@login_required
def list_album():
    dataset = []
    product_list = Products.query.all()
    for product in product_list:
        dataset.append({'productID': product.productID, 'productCode': product.productCode, 'name': product.name,
                       'quantity': product.quantity, 'price': product.price, 'supplierID': product.supplierID})
    return render_template('list_album2.html', entries=dataset)
```

Similar to the SQL “select * from products”

```
@app.route('/album')
def album():
    page_num = 1
    products = Products.query.paginate(per_page=9, page=page_num, error_out=True)
    return render_template('product_paging.html', products=products)
```

Passing how many entries per page (e.g. 9 entries).

```
@app.route('/album/<int:page_num>')
def album_paging(page_num):
    products = Products.query.paginate(per_page=9, page=page_num, error_out=True)
    return render_template('product_paging.html', products=products)
```

```
@app.route("/add_product", methods=['POST'])
@login_required
def add_product():
    productCode = request.form['product_code']
    name = request.form['name']
    quantity = request.form['quantity']
    price = float(request.form['price'])
    supplierID = request.form['supplier id']
    product = Products(productCode=productCode, name=name, quantity=quantity, price=price, supplierID=supplierID)
    db.session.add(product)
    db.session.commit()

    dataset = []
    product_list = Products.query.all()
    for product in product_list:
        dataset.append({'productID': product.productID, 'productCode': product.productCode, 'name': product.name,
                       'quantity': product.quantity, 'price': product.price, 'supplierID': product.supplierID})
    return render_template('list_album2.html', entries=dataset)
```

Add operation

```
@app.route("/update_product",methods=['POST','PUT'])
@login_required
def update_product():
    record_id = request.form['record_id']
    product = Products.query.filter_by(productID=record_id).first()
    product.productCode = request.form['product_code']
    product.name = request.form['name']
    product.quantity = request.form['quantity']
    product.price = float(request.form['price'])
    product.supplierID = request.form['supplier_id']
    db.session.commit()

    dataset = []
    product_list = Products.query.all()
    for product in product_list:
        dataset.append({'productID': product.productID, 'productCode': product.productCode, 'name': product.name,
                       'quantity': product.quantity, 'price': product.price, 'supplierID': product.supplierID})
    return render_template('list_album2.html', entries=dataset)

@app.route("/delete_product",methods=['POST','DELETE'])
@login_required
def delete_product():
    #url = "https://api.airtable.com/v0/appM38HX1EVhxmnax/flaskdemo/"
    record_id = request.form['record_id']
    product = Products.query.filter_by(productID=record_id).first()
    db.session.delete(product)
    db.session.commit()

    dataset = []
    product_list = Products.query.all()
    for product in product_list:
        dataset.append({'productID': product.productID, 'productCode': product.productCode, 'name': product.name,
                       'quantity': product.quantity, 'price': product.price, 'supplierID': product.supplierID})
    return render_template('list_album2.html', entries=dataset)
```

Update operation

Delete operation

**Introducing
Flask
SQLAlchemy**

**CRUD Support in
SQLAlchemy and
Airtable for
View Integration**

**Unit Testing
Integration Testing
User Acceptance
Testing**

Airtable REST API **CRUD** Operations

In [33]:

```
1 #  
2 # List Records  
3 #  
4 import requests  
5 import pandas as pd  
6  
7 headers = {  
8     'Authorization': 'Bearer ' ,  
9 }  
10  
11 r = requests.get('https://api.airtable.com/v0/appM38HX1EVhxmnqx/flaskdemo?sortField=_createdTime&sortDirection=desc')  
12 print("Status Code:",r.status_code)  
13 print(r.text)  
14 dict = r.json()  
15 a_list = dict['records']  
16 df = pd.DataFrame.from_dict(a_list)  
17 df.head()
```

Status Code: 200

```
{"records":[{"id":"recb1ZFR8ZL0HB9ti","fields":{"full_name":"William Lo","lname":"Lo","student_id":"5555","date_of_birth":"2002-02-21","record_id":"recb1ZFR8ZL0HB9ti","fname":"William"},"createdTime":"2019-09-21T02:11:26.000Z"}, {"id":"reczlouWve6Oh1JPv","fields":{"full_name":"Winifred Wong","lname":"Wong","student_id":"010585","date_of_birth":"1958-01-28","record_id":"reczlouWve6Oh1JPv","fname":"Winifred"},"createdTime":"2019-09-20T16:37:52.000Z"}, {"id":"rect6erpDSPNty1TN","fields":{"full_name":"Bernard Suen","lname":"Suen","student_id":"11111","date_of_birth":"1958-05-23","record_id":"rect6erpDSPNty1TN","fname":"Bernard"},"createdTime":"2019-09-19T15:18:41.000Z"}, {"id":"rec8vF2VbIgOTdx3F","fields":{"full_name":"Mary Lu","lname":"Lu","student_id":"99999","date_of_birth":"2001-08-22","record_id":"rec8vF2VbIgOTdx3F","fname":"Mary"},"createdTime":"2019-09-19T14:32:08.000Z"}, {"id":"recASxs4aPaHLeJj3","fields":{"full_name":"Emily Chan","lname":"Chan","student_id":"67891","date_of_birth":"2008-04-03","record_id":"recASxs4aPaHLeJj3","fname":"Emily"},"createdTime":"2019-09-19T14:04:24.000Z"}]}
```

Out[33]:

	createdTime	fields	id
0	2019-09-21T02:11:26.000Z	{'full_name': 'William Lo', 'lname': 'Lo', 'stu...}	recb1ZFR8ZL0HB9ti
1	2019-09-20T16:37:52.000Z	{'full_name': 'Winifred Wong', 'lname': 'Wong'...}	reczlouWve6Oh1JPv
2	2019-09-19T15:18:41.000Z	{'full_name': 'Bernard Suen', 'lname': 'Suen',...}	rect6erpDSPNty1TN
3	2019-09-19T14:32:08.000Z	{'full_name': 'Mary Lu', 'lname': 'Lu', 'stude...}	rec8vF2VbIgOTdx3F
4	2019-09-19T14:04:24.000Z	{'full_name': 'Emily Chan', 'lname': 'Chan', '...}	recASxs4aPaHLeJj3

In [38]:

```
1 #  
2 # Add Record  
3 #  
4  
5 import requests  
6 import pandas as pd  
7  
8 data = {  
9     "records": [  
10         {  
11             "fields": {  
12                 "fname": "Emily",  
13                 "lname": "Lau",  
14                 "student_id": "686868",  
15                 "date_of_birth": "2002-02-21"  
16             }  
17         }  
18     ]  
19 }  
20 headers = {'Authorization': 'Bearer - - - - -', 'Content-Type': 'application/json; charset=utf-8'}  
21 r = requests.post('https://api.airtable.com/v0/appM38HX1EVhxmnqx/flaskdemo', json=data, headers=headers)  
22 print("Status Code:", r.status_code)  
23 print(r.text)  
24 dict = r.json()  
25 a_list = dict['records']  
26 df = pd.DataFrame.from_dict(a_list)  
27 df.head()  
  
Status Code: 200  
{ "records": [{ "id": "recggGoKr4zQhqj3l", "fields": { "lname": "Lau", "student_id": "686868", "date_of_birth": "2002-02-21", "fname": "Emily", "full_name": "Emily Lau", "record_id": "recggGoKr4zQhqj3l" }, "createdTime": "2019-09-21T02:53:23.000Z" } ]}
```

Out[38]:

	createdTime	fields	id
0	2019-09-21T02:53:23.000Z	{'lname': 'Lau', 'student_id': '686868', 'date... recggGoKr4zQhqj3l	

In [40]:

```
1 #  
2 # Update Records  
3 #  
4 import requests  
5 import pandas as pd  
6  
7 data = {  
8     "records": [  
9         {  
10             "id": "recggGoKr4zQhqj3l",  
11             "fields": {  
12                 "fname": "Emily",  
13                 "lname": "Chan",  
14                 "student_id": "67891",  
15                 "date_of_birth": "2008-04-03"  
16             }  
17         }  
18     ]  
19 }  
20 headers = {'Authorization': 'Bearer', 'Content-Type': 'application/json; charset=utf-8'}  
21 r = requests.put('https://api.airtable.com/v0/appM38HX1EVhxmnqx/flaskdemo', json=data, headers=headers)  
22 print("Status Code:", r.status_code)  
23 print(r.text)  
24 dict = r.json()  
25 a_list = dict['records']  
26 df = pd.DataFrame.from_dict(a_list)  
27 df.head()  
  
Status Code: 200  
{"records": [{"id": "recggGoKr4zQhqj3l", "fields": {"lname": "Chan", "student_id": "67891", "date_of_birth": "2008-04-03", "fname": "Emily", "full_name": "Emily Chan", "record_id": "recggGoKr4zQhqj3l"}, "createdTime": "2019-09-21T02:53:23.000Z"}]}
```

Out[40]:

	createdTime	fields	id
0	2019-09-21T02:53:23.000Z	{"lname": "Chan", "student_id": "67891", "date... recggGoKr4zQhqj3l	

In [41]:

```
1 #  
2 # Delete Record  
3 #  
4 import requests  
5 import pandas as pd  
6  
7 headers = {'Authorization': 'Bearer', 'Content-Type': 'application/x-www-form-urlencoded'}  
8 r = requests.delete('https://api.airtable.com/v0/appM38HX1EVhxmnqx/flaskdemo/recggGoKr4zQhqqj31',headers=headers)  
9 print("Status Code:",r.status_code)  
10 print(r.text)  
11  
12 headers = {  
13     'Authorization': 'Bearer',  
14 }  
15 r = requests.get('https://api.airtable.com/v0/appM38HX1EVhxmnqx/flaskdemo?  
16                 sortField =_createdTime&sortDirection=desc', headers=headers)  
17 dict = r.json()  
18 a_list = dict['records']  
19 df = pd.DataFrame.from_dict(a_list)  
20 df.head()  
  
Status Code: 200  
{"deleted":true,"id":"recggGoKr4zQhqqj31"}
```

Out[41]:

	createdTime	fields	id
0	2019-09-21T02:11:26.000Z	{'full_name': 'William Lo', 'Iname': 'Lo', 'stu...}	recb1ZFR8ZL0HB9ti
1	2019-09-20T16:37:52.000Z	{'full_name': 'Winifred Wong', 'Iname': 'Wong'...}	reczlouWve6Oh1JPv
2	2019-09-19T15:18:41.000Z	{'full_name': 'Bernard Suen', 'Iname': 'Suen',...}	rect6erpDSPNty1TN
3	2019-09-19T14:32:08.000Z	{'full_name': 'Mary Lu', 'Iname': 'Lu', 'stude...}	rec8vF2VblgOTdx3F

API End-points for Front-end Access Using JSONIFY

```
@app.route('/api')
def api():
    result = db.engine.execute('select * from products')
    final_result = [list(i) for i in result]
    dataset=[]
    dict={}
    for i in final_result:
        dict['productID'] = i[0]
        dict['productCode'] = i[1]
        dict['name'] = i[2]
        dict['quantity'] = i[3]
        dict['price'] = i[4]
        dict['supplierID'] = i[5]
        # print(i)
        # print(dict)
        dataset.append(dict.copy()) #markers.append(fld.copy())
    return jsonify({'Album': dataset})
```

NEW **Runner** **Import** **Builder** **Team Library** **SYNC OFF** **Sign In**

⚠ Chrome apps are being deprecated. [Download](#) our free native apps for continued support and better performance. [Learn more](#)

Filter

History Collections

All Me Team

JS Ajax REST Get/Post ★ 2 requests

RESTful Test 14 requests

Token Authentication Test 11 requests

GET http://localhost:9009/api/post

POST http://localhost:9009/api/post

POST localhost:9030/api_auth

GET localhost:9030/api_album

GET localhost:9031/login

GET localhost:9031/user

POST localhost:9031/user

GET localhost:9031/todo

GET localhost:9031/user

localhost http://localhost: localhost: localhost: localhost: localhost: localhost: localhost: localhost: http

No Environment

GET http://localhost:9030/api

Authorization Headers Body Pre-request Script Tests

Type No Auth

Body Cookies Headers (5) Test Results Status: 200 OK Time: 169 ms

Pretty Raw Preview JSON

```
1 {  
2   "Album": [  
3     {  
4       "name": "Pen Red",  
5       "price": 1.23,  
6       "productCode": "PEN",  
7       "productID": 1001,  
8       "quantity": 5000,  
9       "supplierID": 501  
10      },  
11      {  
12        "name": "Pen Blue",  
13        "price": 1.25,  
14        "productCode": "PEN",  
15        "productID": 1002,  
16        "quantity": 8000,  
17        "supplierID": 502  
18      }  
19    ]  
20  }  
21 }  
22 }  
23 }  
24 }  
25 }  
26 }  
27 }  
28 }  
29 }  
30 }  
31 }  
32 }  
33 }  
34 }  
35 }  
36 }  
37 }  
38 }  
39 }  
40 }  
41 }  
42 }  
43 }  
44 }  
45 }  
46 }  
47 }  
48 }  
49 }  
50 }  
51 }  
52 }  
53 }  
54 }  
55 }  
56 }  
57 }  
58 }  
59 }  
60 }  
61 }  
62 }  
63 }  
64 }  
65 }  
66 }  
67 }  
68 }  
69 }  
70 }  
71 }  
72 }  
73 }  
74 }  
75 }  
76 }  
77 }  
78 }  
79 }  
80 }  
81 }  
82 }  
83 }  
84 }  
85 }  
86 }  
87 }  
88 }  
89 }  
90 }  
91 }  
92 }  
93 }  
94 }  
95 }  
96 }  
97 }  
98 }  
99 }  
100 }  
101 }  
102 }  
103 }  
104 }  
105 }  
106 }  
107 }  
108 }  
109 }  
110 }  
111 }  
112 }  
113 }  
114 }  
115 }  
116 }  
117 }  
118 }  
119 }  
120 }  
121 }  
122 }  
123 }  
124 }  
125 }  
126 }  
127 }  
128 }  
129 }  
130 }  
131 }  
132 }  
133 }  
134 }  
135 }  
136 }  
137 }  
138 }  
139 }  
140 }  
141 }  
142 }  
143 }  
144 }  
145 }  
146 }  
147 }  
148 }  
149 }  
150 }  
151 }  
152 }  
153 }  
154 }  
155 }  
156 }  
157 }  
158 }  
159 }  
160 }  
161 }  
162 }  
163 }  
164 }  
165 }  
166 }  
167 }  
168 }  
169 }  
170 }  
171 }  
172 }  
173 }  
174 }  
175 }  
176 }  
177 }  
178 }  
179 }  
180 }  
181 }  
182 }  
183 }  
184 }  
185 }  
186 }  
187 }  
188 }  
189 }  
190 }  
191 }  
192 }  
193 }  
194 }  
195 }  
196 }  
197 }  
198 }  
199 }  
200 }  
201 }  
202 }  
203 }  
204 }  
205 }  
206 }  
207 }  
208 }  
209 }  
210 }  
211 }  
212 }  
213 }  
214 }  
215 }  
216 }  
217 }  
218 }  
219 }  
220 }  
221 }  
222 }  
223 }  
224 }  
225 }  
226 }  
227 }  
228 }  
229 }  
230 }  
231 }  
232 }  
233 }  
234 }  
235 }  
236 }  
237 }  
238 }  
239 }  
240 }  
241 }  
242 }  
243 }  
244 }  
245 }  
246 }  
247 }  
248 }  
249 }  
250 }  
251 }  
252 }  
253 }  
254 }  
255 }  
256 }  
257 }  
258 }  
259 }  
260 }  
261 }  
262 }  
263 }  
264 }  
265 }  
266 }  
267 }  
268 }  
269 }  
270 }  
271 }  
272 }  
273 }  
274 }  
275 }  
276 }  
277 }  
278 }  
279 }  
280 }  
281 }  
282 }  
283 }  
284 }  
285 }  
286 }  
287 }  
288 }  
289 }  
290 }  
291 }  
292 }  
293 }  
294 }  
295 }  
296 }  
297 }  
298 }  
299 }  
300 }  
301 }  
302 }  
303 }  
304 }  
305 }  
306 }  
307 }  
308 }  
309 }  
310 }  
311 }  
312 }  
313 }  
314 }  
315 }  
316 }  
317 }  
318 }  
319 }  
320 }  
321 }  
322 }  
323 }  
324 }  
325 }  
326 }  
327 }  
328 }  
329 }  
330 }  
331 }  
332 }  
333 }  
334 }  
335 }  
336 }  
337 }  
338 }  
339 }  
340 }  
341 }  
342 }  
343 }  
344 }  
345 }  
346 }  
347 }  
348 }  
349 }  
350 }  
351 }  
352 }  
353 }  
354 }  
355 }  
356 }  
357 }  
358 }  
359 }  
360 }  
361 }  
362 }  
363 }  
364 }  
365 }  
366 }  
367 }  
368 }  
369 }  
370 }  
371 }  
372 }  
373 }  
374 }  
375 }  
376 }  
377 }  
378 }  
379 }  
380 }  
381 }  
382 }  
383 }  
384 }  
385 }  
386 }  
387 }  
388 }  
389 }  
390 }  
391 }  
392 }  
393 }  
394 }  
395 }  
396 }  
397 }  
398 }  
399 }  
400 }  
401 }  
402 }  
403 }  
404 }  
405 }  
406 }  
407 }  
408 }  
409 }  
410 }  
411 }  
412 }  
413 }  
414 }  
415 }  
416 }  
417 }  
418 }  
419 }  
420 }  
421 }  
422 }  
423 }  
424 }  
425 }  
426 }  
427 }  
428 }  
429 }  
430 }  
431 }  
432 }  
433 }  
434 }  
435 }  
436 }  
437 }  
438 }  
439 }  
440 }  
441 }  
442 }  
443 }  
444 }  
445 }  
446 }  
447 }  
448 }  
449 }  
450 }  
451 }  
452 }  
453 }  
454 }  
455 }  
456 }  
457 }  
458 }  
459 }  
460 }  
461 }  
462 }  
463 }  
464 }  
465 }  
466 }  
467 }  
468 }  
469 }  
470 }  
471 }  
472 }  
473 }  
474 }  
475 }  
476 }  
477 }  
478 }  
479 }  
480 }  
481 }  
482 }  
483 }  
484 }  
485 }  
486 }  
487 }  
488 }  
489 }  
490 }  
491 }  
492 }  
493 }  
494 }  
495 }  
496 }  
497 }  
498 }  
499 }  
500 }  
501 }  
502 }  
503 }  
504 }  
505 }  
506 }  
507 }  
508 }  
509 }  
510 }  
511 }  
512 }  
513 }  
514 }  
515 }  
516 }  
517 }  
518 }  
519 }  
520 }  
521 }  
522 }  
523 }  
524 }  
525 }  
526 }  
527 }  
528 }  
529 }  
530 }  
531 }  
532 }  
533 }  
534 }  
535 }  
536 }  
537 }  
538 }  
539 }  
540 }  
541 }  
542 }  
543 }  
544 }  
545 }  
546 }  
547 }  
548 }  
549 }  
550 }  
551 }  
552 }  
553 }  
554 }  
555 }  
556 }  
557 }  
558 }  
559 }  
560 }  
561 }  
562 }  
563 }  
564 }  
565 }  
566 }  
567 }  
568 }  
569 }  
570 }  
571 }  
572 }  
573 }  
574 }  
575 }  
576 }  
577 }  
578 }  
579 }  
580 }  
581 }  
582 }  
583 }  
584 }  
585 }  
586 }  
587 }  
588 }  
589 }  
590 }  
591 }  
592 }  
593 }  
594 }  
595 }  
596 }  
597 }  
598 }  
599 }  
600 }  
601 }  
602 }  
603 }  
604 }  
605 }  
606 }  
607 }  
608 }  
609 }  
610 }  
611 }  
612 }  
613 }  
614 }  
615 }  
616 }  
617 }  
618 }  
619 }  
620 }  
621 }  
622 }  
623 }  
624 }  
625 }  
626 }  
627 }  
628 }  
629 }  
630 }  
631 }  
632 }  
633 }  
634 }  
635 }  
636 }  
637 }  
638 }  
639 }  
640 }  
641 }  
642 }  
643 }  
644 }  
645 }  
646 }  
647 }  
648 }  
649 }  
650 }  
651 }  
652 }  
653 }  
654 }  
655 }  
656 }  
657 }  
658 }  
659 }  
660 }  
661 }  
662 }  
663 }  
664 }  
665 }  
666 }  
667 }  
668 }  
669 }  
670 }  
671 }  
672 }  
673 }  
674 }  
675 }  
676 }  
677 }  
678 }  
679 }  
680 }  
681 }  
682 }  
683 }  
684 }  
685 }  
686 }  
687 }  
688 }  
689 }  
690 }  
691 }  
692 }  
693 }  
694 }  
695 }  
696 }  
697 }  
698 }  
699 }  
700 }  
701 }  
702 }  
703 }  
704 }  
705 }  
706 }  
707 }  
708 }  
709 }  
710 }  
711 }  
712 }  
713 }  
714 }  
715 }  
716 }  
717 }  
718 }  
719 }  
720 }  
721 }  
722 }  
723 }  
724 }  
725 }  
726 }  
727 }  
728 }  
729 }  
730 }  
731 }  
732 }  
733 }  
734 }  
735 }  
736 }  
737 }  
738 }  
739 }  
740 }  
741 }  
742 }  
743 }  
744 }  
745 }  
746 }  
747 }  
748 }  
749 }  
750 }  
751 }  
752 }  
753 }  
754 }  
755 }  
756 }  
757 }  
758 }  
759 }  
760 }  
761 }  
762 }  
763 }  
764 }  
765 }  
766 }  
767 }  
768 }  
769 }  
770 }  
771 }  
772 }  
773 }  
774 }  
775 }  
776 }  
777 }  
778 }  
779 }  
780 }  
781 }  
782 }  
783 }  
784 }  
785 }  
786 }  
787 }  
788 }  
789 }  
790 }  
791 }  
792 }  
793 }  
794 }  
795 }  
796 }  
797 }  
798 }  
799 }  
800 }  
801 }  
802 }  
803 }  
804 }  
805 }  
806 }  
807 }  
808 }  
809 }  
810 }  
811 }  
812 }  
813 }  
814 }  
815 }  
816 }  
817 }  
818 }  
819 }  
820 }  
821 }  
822 }  
823 }  
824 }  
825 }  
826 }  
827 }  
828 }  
829 }  
830 }  
831 }  
832 }  
833 }  
834 }  
835 }  
836 }  
837 }  
838 }  
839 }  
840 }  
841 }  
842 }  
843 }  
844 }  
845 }  
846 }  
847 }  
848 }  
849 }  
850 }  
851 }  
852 }  
853 }  
854 }  
855 }  
856 }  
857 }  
858 }  
859 }  
860 }  
861 }  
862 }  
863 }  
864 }  
865 }  
866 }  
867 }  
868 }  
869 }  
870 }  
871 }  
872 }  
873 }  
874 }  
875 }  
876 }  
877 }  
878 }  
879 }  
880 }  
881 }  
882 }  
883 }  
884 }  
885 }  
886 }  
887 }  
888 }  
889 }  
890 }  
891 }  
892 }  
893 }  
894 }  
895 }  
896 }  
897 }  
898 }  
899 }  
900 }  
901 }  
902 }  
903 }  
904 }  
905 }  
906 }  
907 }  
908 }  
909 }  
910 }  
911 }  
912 }  
913 }  
914 }  
915 }  
916 }  
917 }  
918 }  
919 }  
920 }  
921 }  
922 }  
923 }  
924 }  
925 }  
926 }  
927 }  
928 }  
929 }  
930 }  
931 }  
932 }  
933 }  
934 }  
935 }  
936 }  
937 }  
938 }  
939 }  
940 }  
941 }  
942 }  
943 }  
944 }  
945 }  
946 }  
947 }  
948 }  
949 }  
950 }  
951 }  
952 }  
953 }  
954 }  
955 }  
956 }  
957 }  
958 }  
959 }  
960 }  
961 }  
962 }  
963 }  
964 }  
965 }  
966 }  
967 }  
968 }  
969 }  
970 }  
971 }  
972 }  
973 }  
974 }  
975 }  
976 }  
977 }  
978 }  
979 }  
980 }  
981 }  
982 }  
983 }  
984 }  
985 }  
986 }  
987 }  
988 }  
989 }  
990 }  
991 }  
992 }  
993 }  
994 }  
995 }  
996 }  
997 }  
998 }  
999 }
```

```
document).ready(function(){
var counter = 1;
var at_getapi = "https://api.airtable.com/v0/appKIU0zkHt3AVTL/Venues?api_key=keycj6dRwXwYLEjiv";
$.getJSON(at_getapi, function(result) {
$.each(result.records, function(key,value) {
    var name = value.fields.Name;
    var image = value.fields.img_url;
    var org_desc = value.fields.desc;
    var desc = org_desc.substring(0, 300);
    var n = desc.lastIndexOf(" ");
    var desc = desc.substring(0, n)+"...";
    // alert(value.fields.Name);
    if (counter == 1) {$("#album_items").append(`<div class="row">`);}
    $("#album_items").append(`<div class="col-md-4">
<div class="thumbnail">
    
    <div class="caption">
        <h3>${name}</h3>
        <p><!-- ${desc} --></p>
        <button class="btn btn-lg btn-primary" onclick="modal_handle('${name}', '${org_desc}', ${image}, ${n})">
            data-id="107" data-body="testing...">View</button>
    </div>
</div>
</div>`);
```

```
<link href="css/fontawesome.min.css" rel="stylesheet" type="text/css">
<link rel="stylesheet" href="css/style.css">
<script>
    function modal_handle(name, org_desc, image) {
        $("h3.modal-title").html(`${name}`);
        $(".modal-body").html(`<p>${org_desc}</p>`);
        $('#popup-photo').modal('show');
    }

    function topFunction() {
        $('html, body').animate({scrollTop : 0},800);
        return false;
    }
</script>
```

```
$.getJSON(at_getapi, function(result) {
    $.each(result.Album, function(key,value) {
        var productID = value.productID;
        var productCode = value.productCode;
        var name = value.name;
        // var quantity = value.quantity;
        // var price = value.price;
        // var supplierID = value.supplierID;
        // alert(value.fields.Name);
        if (counter == 1) {$("#album_items").append(`<div class="row">`);}
        $("#album_items").append(`<div class="col-md-4">
<div class="thumbnail">
    <div class="caption">
        <h3>${productID}</h3>
        <p><!-- ${productCode} --></p>
        <button class="btn btn-lg btn-primary" onclick="modal_handle('${productID}', '${productCode}', '${name}')"
            data-id="107" data-body="testing...>View</button>
    </div>
</div>
</div>`);
```

|

```
        counter = counter +1; // counte
    
```

```
    <script>
        function modal_handle(productID, productCode, name) {
            $("h3.modal-title").html(` ${productID}`);
            $(".modal-body").html(`<p>${productCode}</p>
<p>${name}</p>`);
            $('#popup-photo').modal('show');
        }

        function topFunction() {
            $('html, body').animate({scrollTop : 0},800);
            return false;
        }
    </script>
```

Home

Album



HKU

[View](#)



HK Science Musuem

[View](#)



Victoria Peak

[View](#)



^

1001[View](#)**1002**[View](#)**1003**[View](#)**1004**[View](#)**1006**[View](#)**1007**[View](#)**1008**[View](#)**1009**[View](#)**1010**[View](#)**1011**[View](#)**1012**[View](#)**1013**[View](#)

**Introducing
Flask
SQLAlchemy**

**CRUD Support in
SQLAlchemy and
Airtable for
View Integration**

**Unit Testing
Integration Testing
User Acceptance
Testing**

Unit Testing, Integration Testing, and User Acceptance Testing (UAT)



敏捷开发中用户故事地图(User Story Mapping)学习

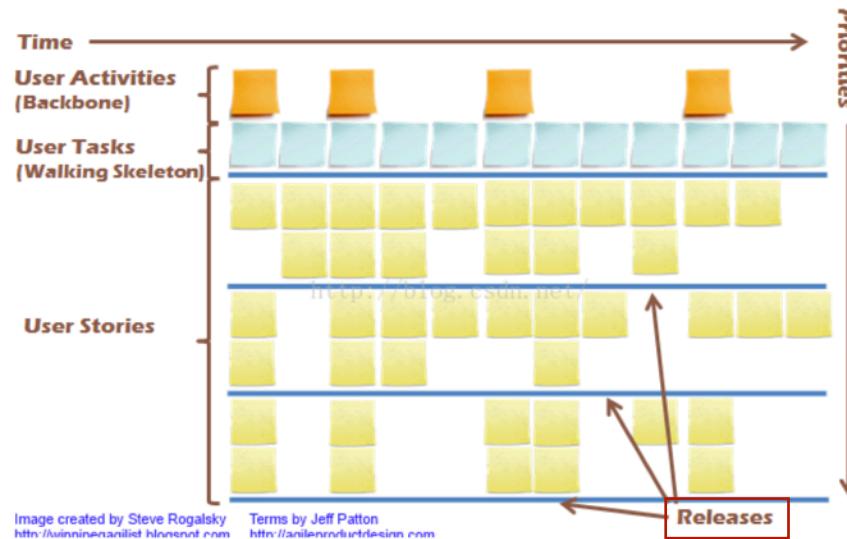
2016年09月25日 20:07:11 [佚名-2018_](#) 阅读数: 3485

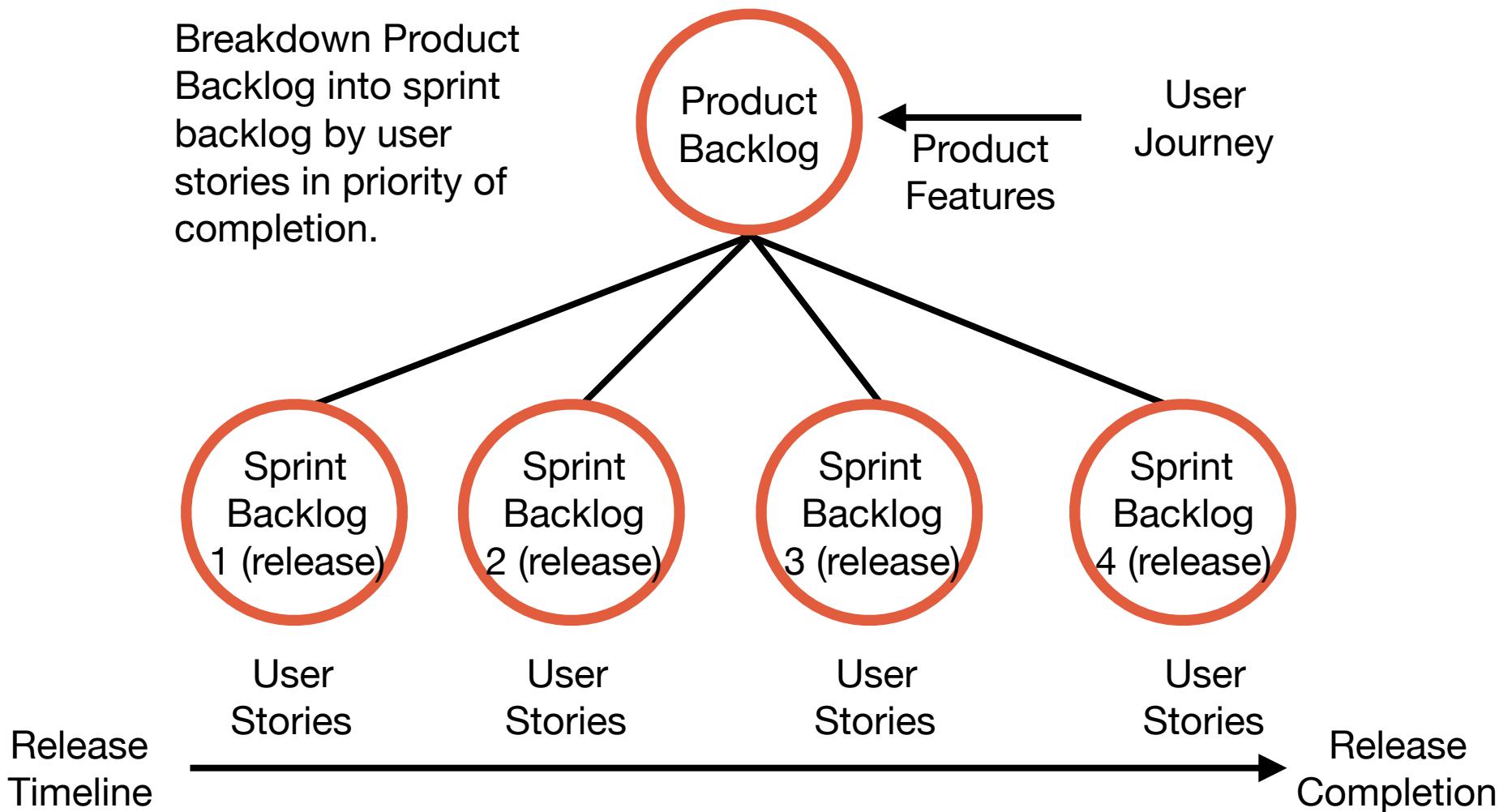
1. 用户故事地图

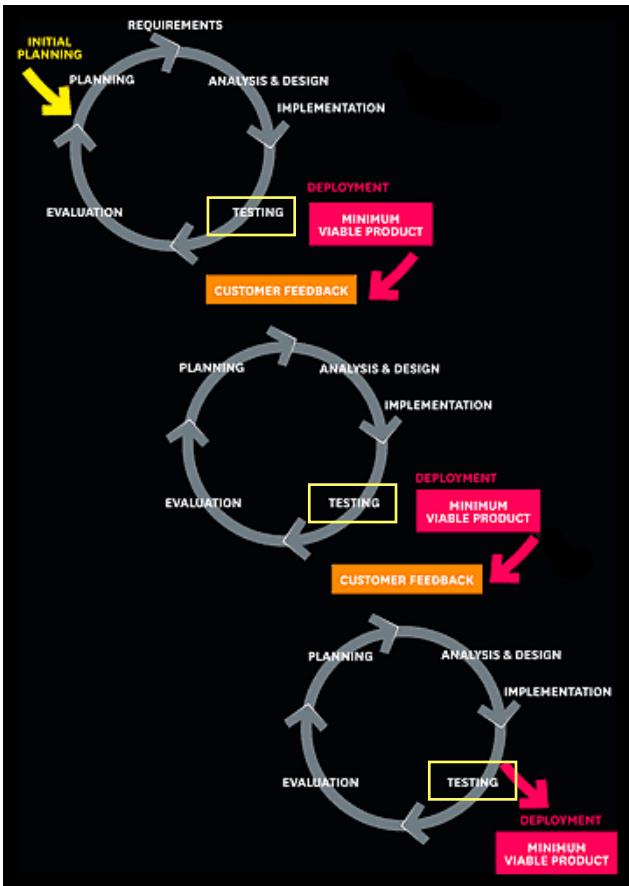
User Story是敏捷开发和管理的核心，User Story是从用户的角度对系统的某个功能模块所作的简短描述。一个User Story描述了项目中的一个小功能，以及这个功能完成之后将会产生什么效果，或者说能为客户创造什么价值。

用户故事地图，英文名为User Story Mapping。用户故事地图已经成为敏捷需求规划中的一个流行方法。用户故事地图可以将你的 backlog（待办事项）变成一张二维地图，而不是传统的简单列表。

2. 用户故事地图编写模板







Source: "Why Lean Start-up Changes Everything"
by Steve Blank

Sprint Planning (Release Planning)



Sprint Review (Release Review)

Daily Scrum (Daily Meeting)

Each build-measure-learn cycle (ends with a review) is called a **"sprint"** usually measured in 1 to 3 weeks.

Testing is an integral part of this process.

Using Stories to Drive the Process



Easy Agile User Story Maps Demo

JIRA Dashboards Projects Issues Boards Create Search

ATV Story Map 6 epics

QUICK FILTERS: Hide Epics openSprints Third-party Unestimated Recently Updated

ISSUES WITHOUT EPICS

ATV-112 Onboarding Create story

ATV-93 Find Movie Create story

ATV-94 Buy Movie Create story

ATV-95 Watch Movie Create story

Issues: 9 Completed: 2 Unestimated: 2 Estimate: 14

ATV-121 Customer Support Create story

ATV-96 Recommend & Refer Create story

Version 2.0 6 5 3 9 of 11 issues visible

ATV-36 Facebook login action - update OAuth token

ATV-1 iTunes user / search by genre 3

ATV-10 iTunes user / pay with connected CC 5

ATV-82 streaming buffer size stalls on HD movies

ATV-122 User / submit contextual support request in-app 5

ATV-31 Rotten Tomatoes ratings not appearing for German users (and Swiss)

ATV-24 user / search by free text (255 character limit) 3

ATV-79 Blank screen on load

ATV-2 user / add Twitter share movie action 5

> Version 4.0 18 9 3 4 issues

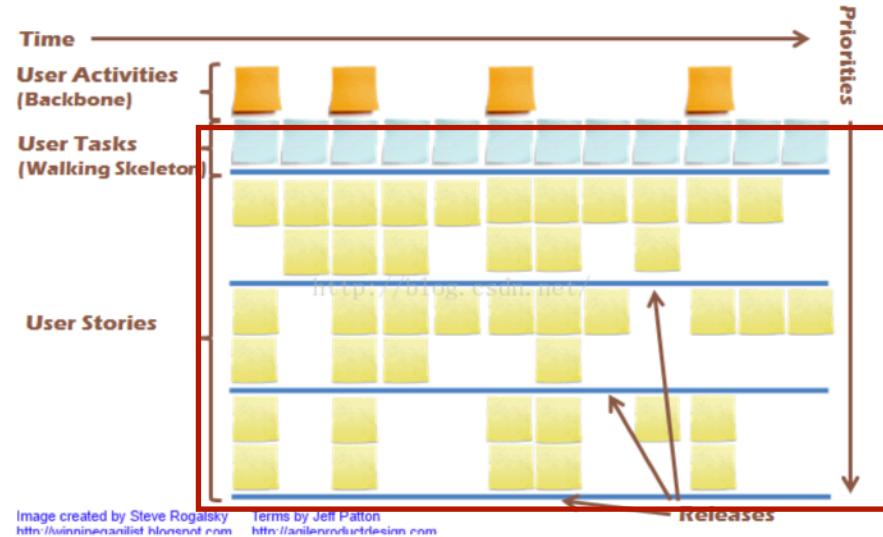
Version 4.1 9 9 6

Version 4.2 9 9 6

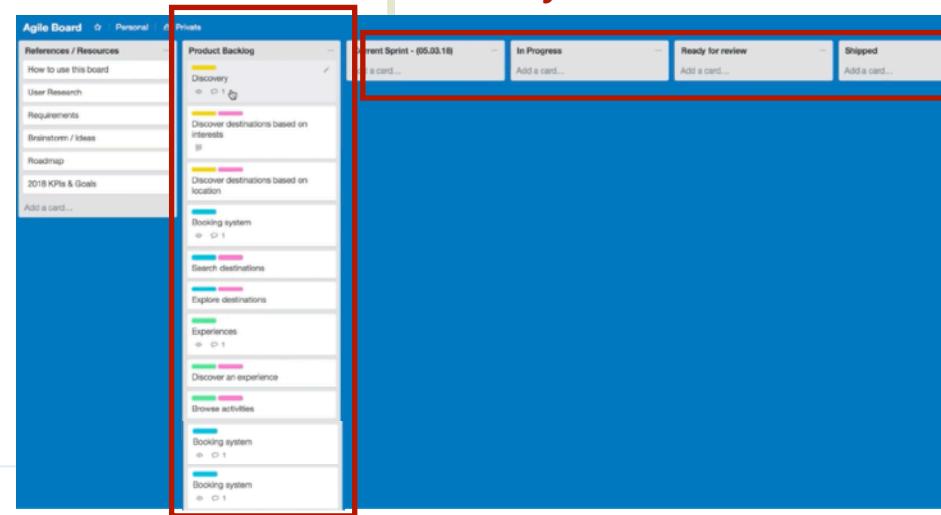
The screenshot shows a JIRA Agile User Story Map interface. At the top, there's a navigation bar with links for JIRA, Dashboards, Projects, Issues, Boards, Create, Search, and various icons for feedback, create epic, filter, help, settings, and user profile. The main title is "ATV Story Map" with "6 epics". Below this, there are "QUICK FILTERS" for Hide Epics, openSprints, Third-party, Unestimated, and Recently Updated. A sidebar on the left lists "ISSUES WITHOUT EPICS" with icons for each category: Onboarding, Find Movie, Buy Movie, Watch Movie, Customer Support, and Recommend & Refer. The "Watch Movie" epic is expanded, showing its details: Issues: 9, Completed: 2, Unestimated: 2, and Estimate: 14. Below the epic, there are several stories: "Facebook login action - update OAuth token" (ATV-36), "iTunes user / search by genre" (ATV-1), "iTunes user / pay with connected CC" (ATV-10), "streaming buffer size stalls on HD movies" (ATV-82), "User / submit contextual support request in-app" (ATV-122), "Rotten Tomatoes ratings not appearing for German users (and Swiss)" (ATV-31), "user / search by free text (255 character limit)" (ATV-24), "Blank screen on load" (ATV-79), and "user / add Twitter share movie action" (ATV-2). Each story has a small icon (red square for tasks, green square for stories, blue square for bugs, yellow square for user stories, and grey square for others) and a number indicating its priority or count. Below the stories, there are sections for "Version 4.0" (4 issues), "Version 4.1" (6 issues), and "Version 4.2" (6 issues).

[Download t.co/getusm](https://t.co/getusm)

2. 用户故事地图编写模板



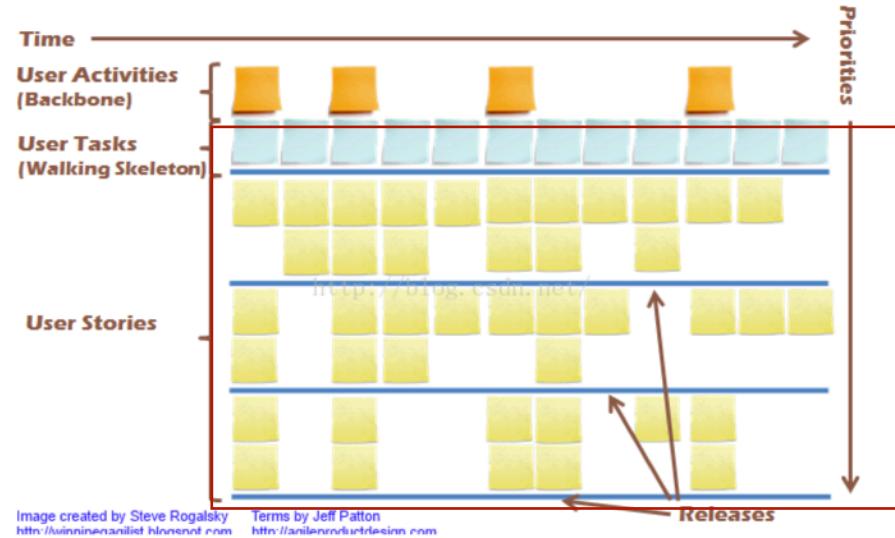
Mapping the backlog and sprinting tasks to the Kanban board to see how the project status changes over time.



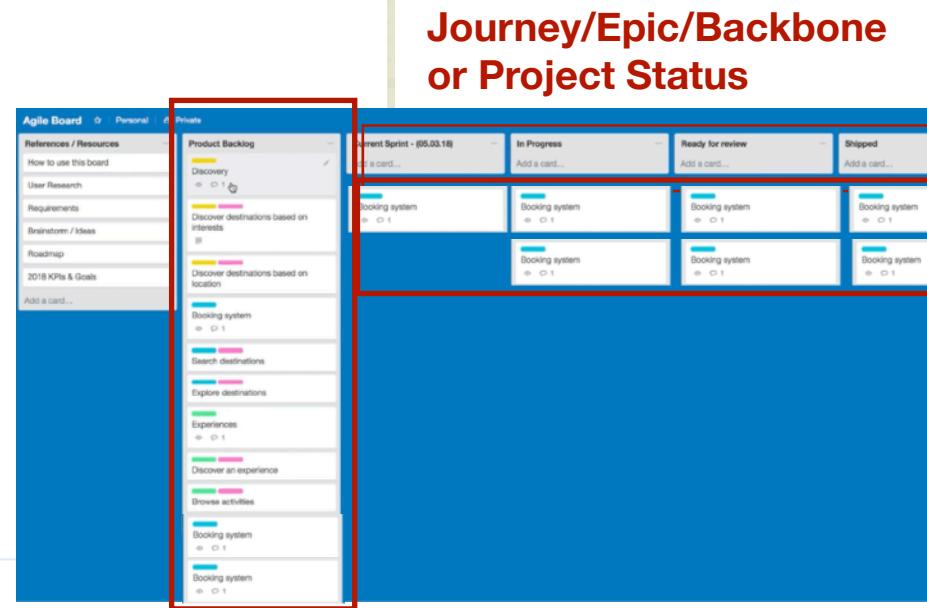
Backlog Made Up of User Stories

Kanban View by Journey Stage/Epic/Backbone

2. 用户故事地图编写模板



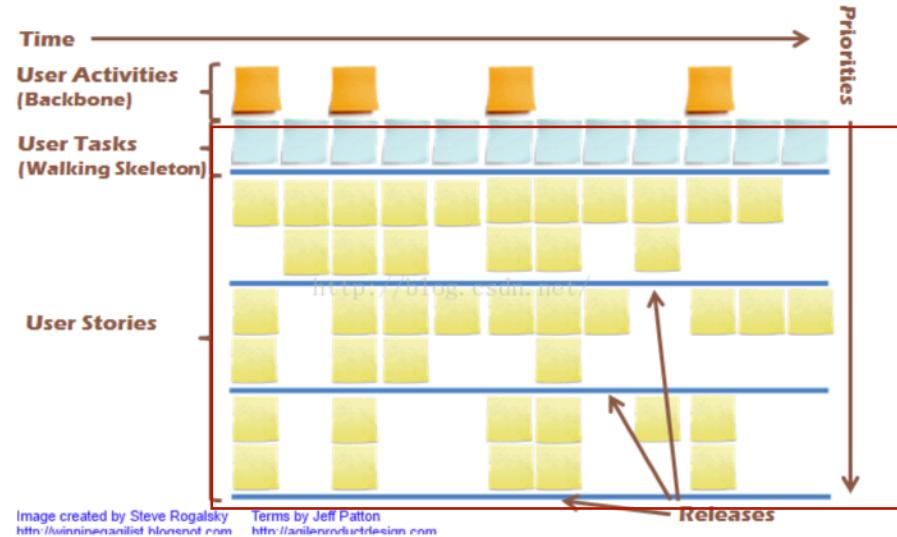
Mapping the backlog and sprinting tasks to the Kanban board to see how the project status changes over time.



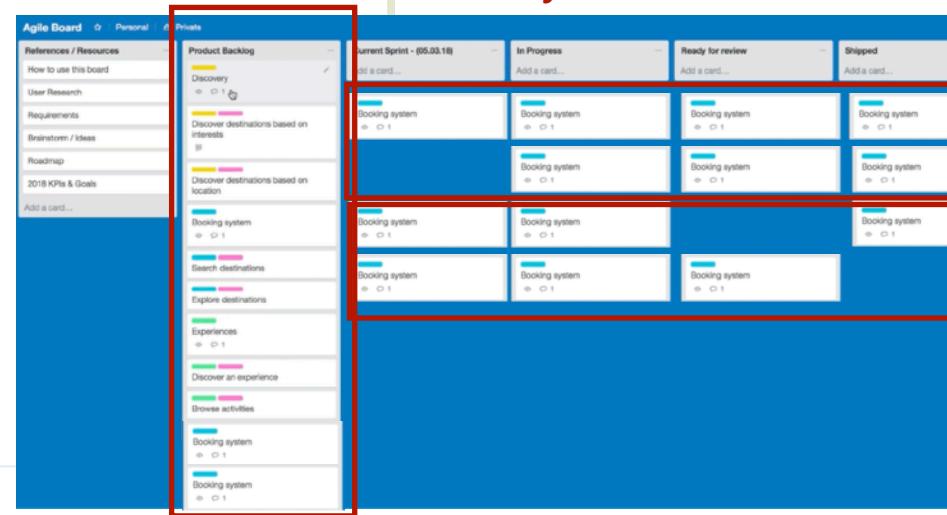
Journey/Epic/Backbone or Project Status

Backlog Made Up of User Stories

2. 用户故事地图编写模板



Mapping the backlog and sprinting tasks to the Kanban board to see how the project status changes over time.



Backlog
Made Up
of User
Stories

Journey/Epic/Backbone
or Project Status

Team Backlog Sprints user study Team Members user journey workflow 1.0 usability test 2.0 Usability Test 用户 会员特权

Kanban Stacked by Status Customize cards Filter Sort Color ...

Uncategorized	Not Started	Assigned	In progress	Finished
No records	No records	No records	No records	wirerrame prototyping : https://marvelapp.com/project/3817331/
+				

When the project completes, all the tasks will be marked “finished” sitting on the far right of the Kanban.

→

1.4 - Test

NOTES

* task based usability test :
1. 找家附近的餐厅去吃
2. 发布一篇食评
3. 收藏一家餐厅到【适合聚餐】 ...

SPRINT

Sprint 2: Low-fi (1.0+2.0)

TEAM MEETING

5/3/2019

SPRINT DAYS LEFT

-357

SPRINT

Sprint 2: Low-fi (1.0+2.0)

TEAM MEETING

12/3/2019

SPRINT DAYS LEFT

-357

25 records [+](#)

Building Different Views to Monitor the Project

Templates

Use these starter bases to get a jump start on your project. For inspiration from community-published bases, check out Universe.



Search templates

CATEGORIES

- Featured
- Content production
- Creative
- Event Planning
- Everyday Life
- Groups, Clubs & Hobbies
- HR & Recruiting
- Legal
- Local Business
- Marketing
- Nonprofit
- Personal
- PR & Communications
- Product, design, and UX
- Project Management
- Publishing
- Real Estate

Agile Workflow

Use template

#PRODUCT, DESIGN, AND UX

Even if you don't want to adopt the entire Agile manifesto, shifting your workflow to incorporate aspects of the Agile methodology—a small-a "agile" workflow—is a good idea for virtually any software development team.

By taking an agile approach to the development process, you let customer needs direct the features and improvements you're working on. By tailoring your product development to focus on the user, you waste less effort on irrelevant or unnecessary features than you do with traditional approaches.

Our blog post here walks you through how to use this template to apply agile processes to suit your team's needs. Hand it off to the scrum master to build your next sprint, schedule standups for your team, and make a note of stories for future sprints in one database that is easily shareable between your entire team. By tagging every potential feature or improvement in a project with the function it serves, the utility it provides to the customer, and its prioritization in the product, this base helps you gather steam and keep rolling into the next sprint after each completed project.

Backlog	Sprints	Standups	People
Main Story View	Hide fields	Filter	Group Sort ...
Storylines	Priority	Duration	Sprint Status

Adding Testing into the Picture

Agile Workflow

	Story Name	Priority	Function	Sprint	Status	Use Case	Notes	story_id	Sprint Release Date
1	S1 : Remote login	★★★	Remote login	Mobile	In Progress		I can login to the app from where...	1	2/19/2018
2	S2 : SSO	★★★★★	SSO	Mobile	Done (Needs Review)		I can use my Google account to lo...	2	2/19/2018
3	S3 : More color options	★★★★★	More color options	UI	Not Started		I can customize the colors to my li...	3	3/14/2018
4	S4 : Customizable referral link	★★★	Customizable referral link		Not Started		I can send more effective referral ...	4	
5	S5 : Zapier integration	★★★★★	Zapier integration	Integrations	Finished		I can build 3rd-party utilities usin...	5	2/1/2018
6	S6 : Bitcoin support for transactio...	★★★	Bitcoin support for transactions	Integrations	Finished		I can spend my BTC with your app	6	2/1/2018
7	S7 : Night reading mode	★★★★★	Night reading mode	UI	Not Started		I don't hurt my eyes using the app...	7	3/14/2018
8	S16 : Multiple timezone support	★★	Multiple timezone support		Not Started		I can use the app from around the...	16	
9	S17 : Cleaned up mobile icon	★★★	Cleaned up mobile icon	UI	Not Started		I can enjoy looking at the app on ...	17	3/14/2018
10	S18 : Export as PDF option	★★★	Export as PDF option		Not Started		I can send content to my boss thr...	18	
11	S19 : Import feature	★★	Import feature		Not Started		I can add my own content to the a...	19	
12	S20 : Search feature	★★★★★	Search feature				I can more easily find the pages I ...	20	
13	S21 : Search bar on homepage	★★★★★	Search bar on homepage				I can more easily find the pages I ...	21	

Each unit of work can be separately performed by one to two team members.
Using the MVC framework to further break down the works is a common approach
with some people working on the views (UI) while others working on the back-end
preparing the data model, routes, API endpoints, and processing logic.

In some cases, there will be a content team busy acquiring and validating data to be ready for the database. All these works require testing, documentation and integration support. During more mature releases, users will be brought in for UAT.



Integration Testing Approaches

Top-Down Approach(TDA)

Testing takes place from top to bottom, following the control flow or architectural structure

Bottom-UP Approach(BUA)

Testing takes place from the bottom of the control flow upwards

THANK YOU FOR YOUR TIME!

ASSIGNMENT #5

1. Change the **MySQL** connection module to **Flask-SQLAlchemy**.
2. Add **pagination** and **search** support to the album.
3. Add a **user table** to the album application with **CRUD** support to maintain a list of **authenticated** users for accessing the album.
4. **Add CRUD** support to the “**album**” **table** so new album entries can be added and old ones be updated and even deleted if needed (such **rights** are **only granted** to **admin** user).
5. Develop a mobile front-end based on the **REST API** for user to view the album and authenticated by the **JWT** method. User can change one’s profile if needed (**optional**)



Flask Whoosh *Alchemy*

Flask-WhooshAlchemy is a Flask extension that integrates the text-search functionality of [Whoosh](#) with the ORM of [SQLAlchemy](#) for use in [Flask](#) applications.

Source code and issue tracking at [GitHub](#).

Install

```
pip install flask_whooshalchemy
```

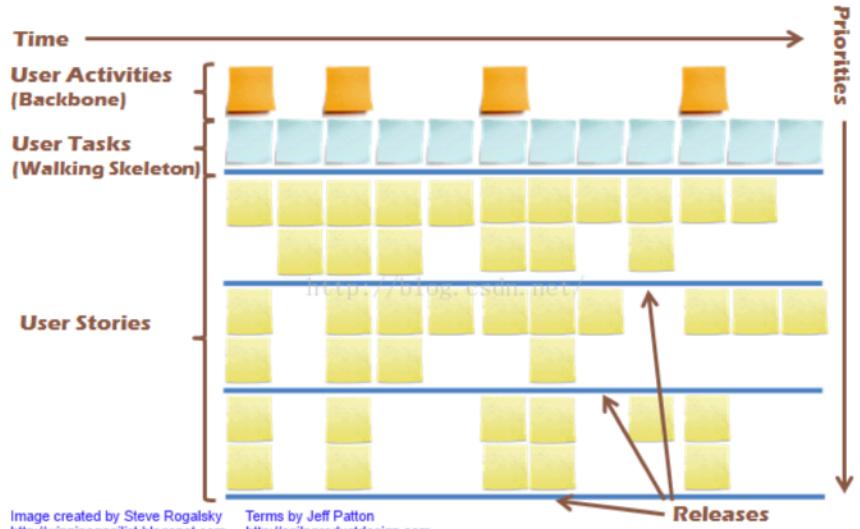
Or:

```
git clone https://github.com/gyllstromk/Flask-WhooshAlchemy.git
```

Quickstart

Let's set up the environment and create our model:

<https://pythonhosted.org/Flask-WhooshAlchemy/>



Discover destinations based on interests

Secure | https://trello.com/c/0i0rV3vL/4-discover-destinations-based-on-interests

Agile Board

References / Resources

- How to use this board
- User Research
- Requirements
- Brainstorm / Ideas
- Roadmap
- 2018 KPIs & Goals

Add a card...

Product Backlog

- Discovery
- Discover destinations based on interests
- Discover destinations based on location
- Booking system
- Search destinations
- Explore destinations
- Experiences
- Discover an experience
- Browse activities

Current S...

Discover destinations based on interests

in list Product Backlog

Labels

Feature 1 User Story +

Description

Edit

User Story: As a user I want to be recommended destinations based on my interests so I can discover new destinations

Add Comment

Write a comment...

Save

Activity

Show Details

Add

- Members
- Labels
- Checklist
- Due Date
- Attachment

Actions

- Move
- Copy
- Subscribe
- Archive

Shipped

Add a card...

... Show Menu

 Reload bssuen.pythonanywhere.com

Best before date:

We're happy to host your free website – and keep it free – for as long as you want to keep it running, but you'll need to log in at least once every three months and click the "Run until 3 months from today" button below. We'll send you an email a week before the site is disabled so that you don't forget to do that. [See here for more details.](#)

This site will be disabled on **Monday 18 May 2020**

 Run until 3 months from today

Paying users' sites stay up forever without any need to log in to keep them running.

Traffic:

How busy is your site?

This month (previous month)	357	(26)
Today (yesterday)	0	(0)
Hour (previous hour)	0	(0)

Want some more data? [Paying accounts get pretty charts ;\)](#)

Code:

What your site is running.

Source code:	/home/bssuen/mysite	 Go to directory
Working directory:	/home/bssuen/	 Go to directory
WSGI configuration file:	/var/www/bssuen_pythonanywhere_com_wsgi.py	
Python version:	3.7 	

Virtualenv:

Use a virtualenv to get different versions of flask, django etc from our default system ones. [More info here](#). You need to [Reload your web app](#) to activate it; NB - will do nothing if the virtualenv does not exist.

[Enter path to a virtualenv, if desired](#)

Log files:

The first place to look if something goes wrong.

Access log:	bssuen.pythonanywhere.com.access.log
Error log:	bssuen.pythonanywhere.com.error.log
Server log:	bssuen.pythonanywhere.com.server.log

Log files are periodically rotated. You can find old logs here: [/var/log](#)