Summary

This is the intial project to kickstart the LoanKit version 3.0 (LK3) development. Please read the instructions below to get started

1. Installation

You will need to checkout the project, setup a virtual environment and import all the requirements to get started. Follow the steps below to do this.

If you are on windows, make sure to download the Python 3.6 installer from the python.org website.

```
# Checkout the project
git clone #TBA - not abailable at time of writing

# Using the python virtual environment module (venv), create a python
# environment into "./env"
python3.6 -m venv env

# Setup your shell for this project
source env/bin/activate

# Or if your on windows
./env/Scripts/activate.bat

# Instal requirements for this project
pip install -r requirements.txt
```

Your now set to go and start developing!

2. Quick Start

Basic Site Setup

At the base of the project, you'll find a `./site` folder. This is where the website lives and breaths. Go into it and create a folder `./site/quick_start/`.

Now we just need to create our first, and simple page, into your folder create the file `index.py` and add the following content.

```
from simplerr.web import web

@web('/')
def echo(request):
    return "Hello from quick_start application"
```

Notice that the function takes a ‘request’ object as it’s first parameter. This variable is where you can access query string and for variables using `request.args['var']` and `request.form['var']` respectively.

Now let’s run it from our web server using the manage.py utility.

```
./manage.py runserver --site ./site/quick_start/
```

You should now be able to browse to http://localhost:5000/ - Congratulations on your first page!

More on routes

Routes in this framework are pretty flexible, you can add various enpoints such as ‘/home’, ‘/user’, etc and server relevant content from it.

Accepting data from routes, query strings, and forms

Add the following code to index.py, these demonstrate some of the various methods to take inputs and ata.

```
@web('/echo/<msg>')
def echo(request, msg):
    return "Echo from page: {}".format(msg)

@web('/echo/args')
def echo_args(request):
    return "Echo using args: {}".format(request.args['msg'])

@web('/echo/form')
def echo_form(request):
    msg = "NOTHING"

    if "msg" in request.form.keys():
        msg = request.form["msg"]

    return """
    <html>
    <body>
     <form method=post action="">
      Mesage: <input type=text name="msg" value="" placeholder="Enter msg val
ue"/><input type="submit">
      You typed in: "{}"
     </form>
    </body>
    </html>
    """.format(msg)
```

Try and browse to the following locations

- http://localhost:5000/echo/Hello World
- http://localhost:5000/echo/args?msg=Hello World
- http://localhost:5000/echo/form

Sending out json for those delicious ajax requests

Ok, so we can send text, what about data for our front end interation? The system is built to detect dictionary, and array returns and convert them to ajax responses.

```
@web('/echo_json/<msg>')
def echo_json(request, msg):
    return {'msg': msg}
```

What about templates

Ok, so you want to render some templates. Easy! Just return a dictionary, or array object and specify the template name as the second parameter to @web()

```
@web('/echo/template/<msg>', 'echo.html')
def echo_template(request, msg):
    return {'msg': msg}
```

Don’t forget to create the template file echo.html

```
Echo: {{msg}}
```

What about database connections

In quick_start, create a model.py file with the following code.

```
from peewee import *

db = SqliteDatabase('./site/quick_start/people.db')

class Person(Model):
    firstname = CharField()
    surname = CharField()

    class Meta:
        database = db

if __name__ == "__main__":
    db.connect()
    db.create_tables([Person])

    # Add some data
    Person(firstname="John", surname="Bob").save()
    Person(firstname="Jane", surname="Bob").save()
    Person(firstname="Michael", surname="Clark").save()
```

To initialise the database, simply run the file using `python ./site/quick_start/modle.py`

And to your index.py file, at the top add a line `from model import Person`

You can now access the database using the following route, and template.

```
@web('/person/api') # This will just return the json for the Persons collecti
on
def person_api(request):
    return Person.select()

@web('/first', template="first.html") # Get the first person in the database
def person_first(request):
    return Person.select().get()
```