Stovie Tech

Temperature monitoring system

Group Info

- Mirza Ishraq Yeahia - Back-end Developer & AWS Administrator
- Kevin Hernandez - Electronics Lead & Raspberry Pi Administrator/Programmer
- Dennis Puasirirutskul - AWS Administrator & UI/UX Designer.
- Nico Rome - Web Developer & Back-end Developer
- Brian Chu - Web & AWS Developer. UX/UI Design
- Arman Tosoonian - Project Manager & Technical Writer
- Aramayis Kageorgis - Backend Developer & Database Administrator

Problem Statement

Families are losing their homes due to ranges or cooktop fires.

Ranges or cooktops were involved in 62% of reported home cooking fires.

What are cooktop manufacturing companies doing to prevent this?

- LED light to indicate when a cooktop is hot.
- WiFi enabled feature to send notifications to a phone (IoT)
- Timing coils to shut off cooktops after a certain amount of time

This is all implemented in some newer models...

What are they doing for old ranges and cooktops?

Manufacturing companies such as LG, Samsung, GE, etc. consider older models as obsolete. They do not offer anything to alleviate this growing issue. They suggest you purchase a new range or cooktop.

Not everyone has the luxury of purchasing a new range or cooktop...

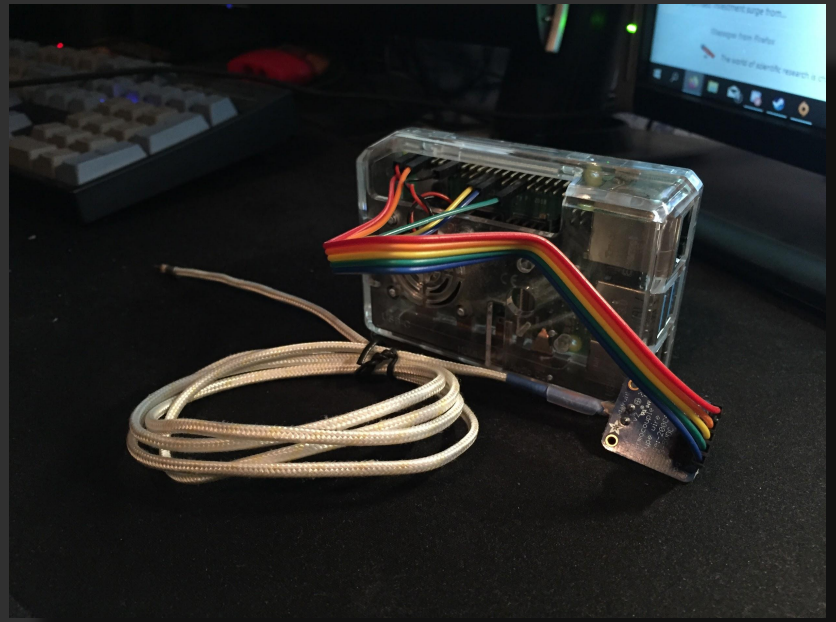
Our Solution

Making your obsolete range or cooktop modern with an IoT device.

With Stovie Tech’s IoT device connected to a range or cooktop, you can get notifications of when the fire is on and also monitor your fire usage through temperatures.

Stovie Tech

Prototype version 0.2



Project Budget

~$300
~100+ Labor hours

The Hardware

- Raspberry Pi 4
- Adafruit MAX31855 Amplifier
- K-Type Thermocouple

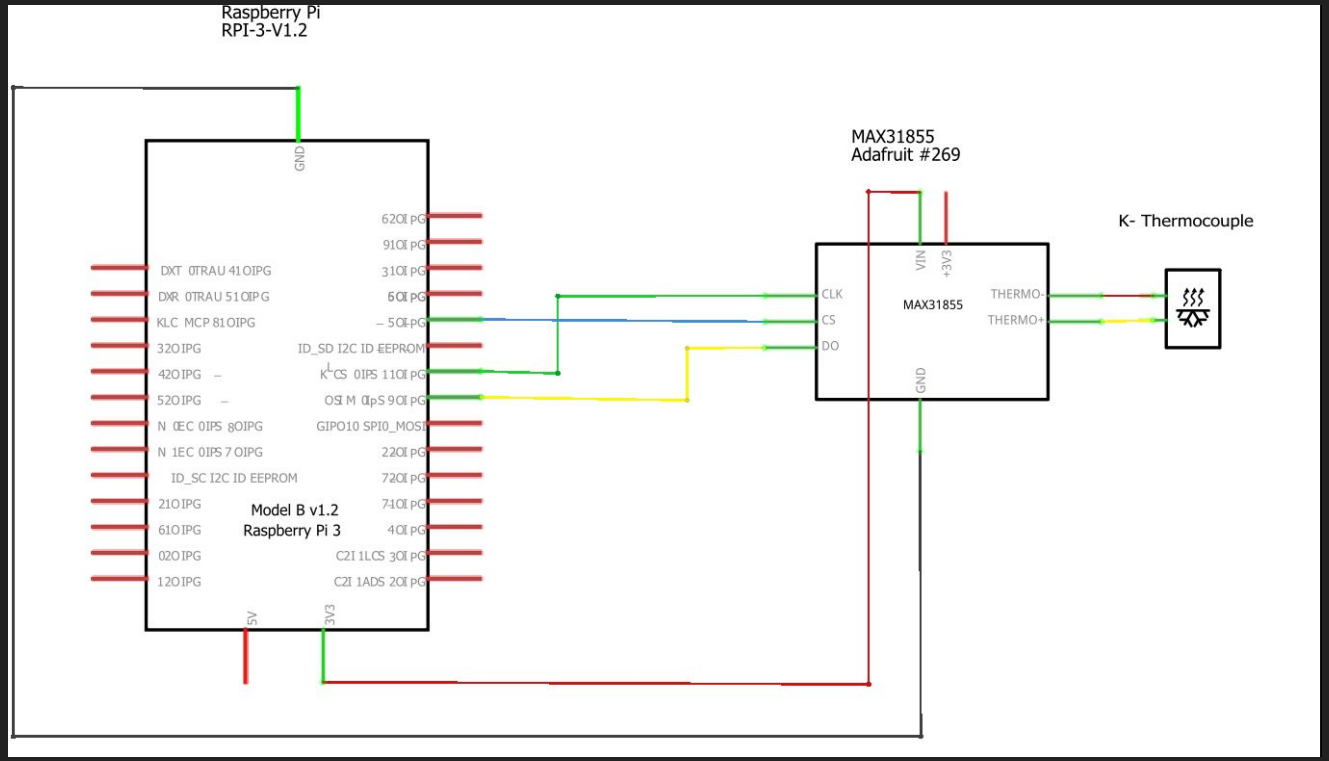
The Software

- Amazon Web services
- AWS S3 bucket
- AWS RDS DB (Postgresql)
- AWS CloudFront
- NameCheap DNS

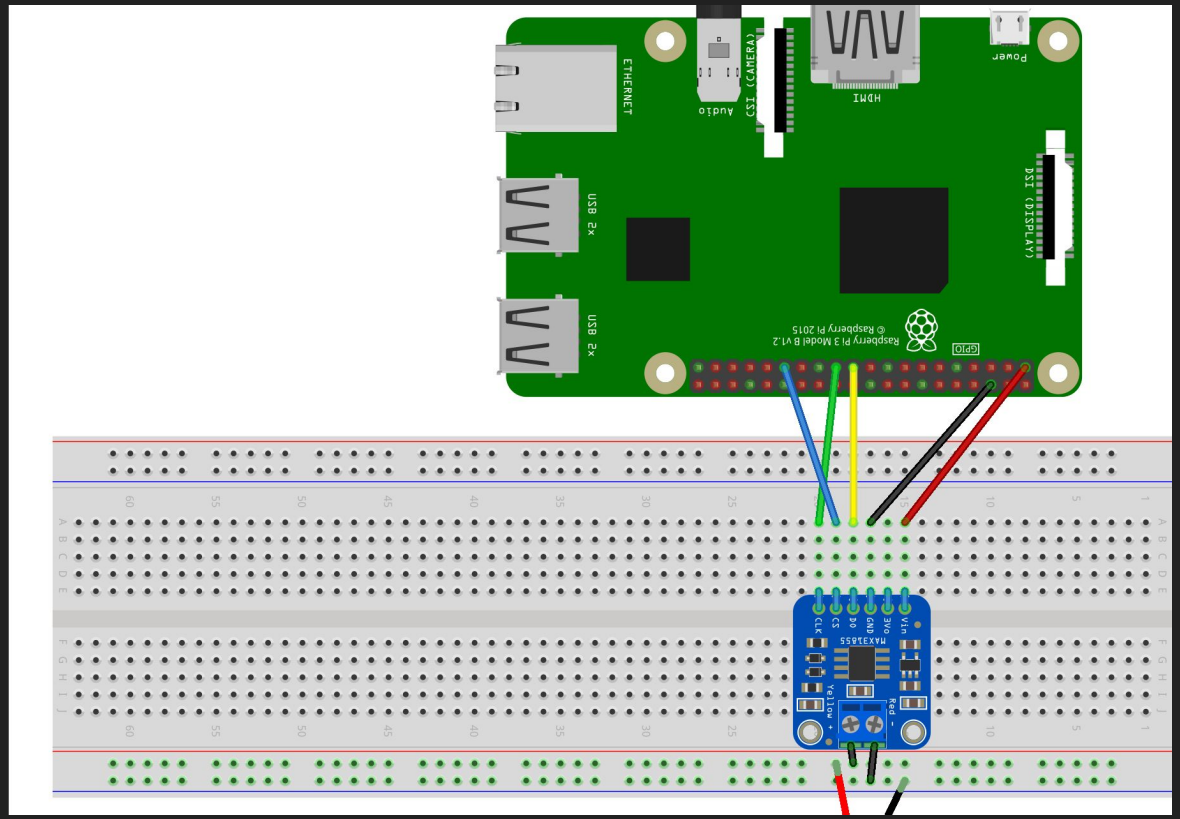
Other

- Portable Stove (testing/demo)

RPi + MAX31855 + Thermocouple Schematic

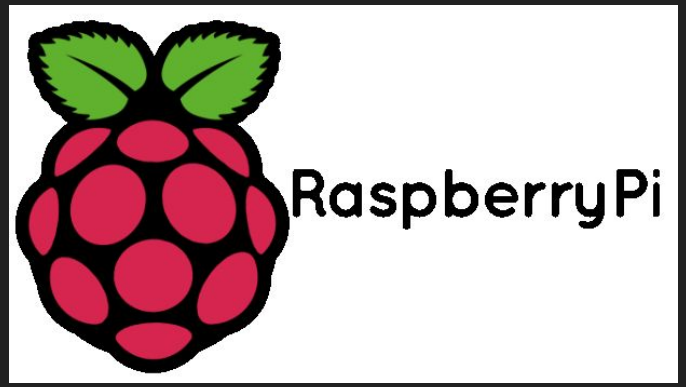


Breadboard Prototype Schematic

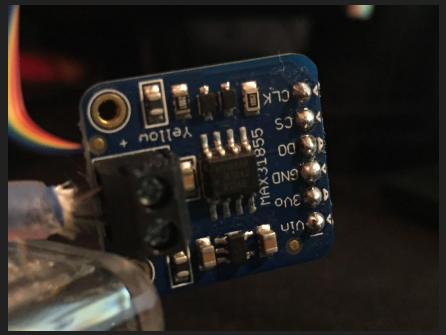


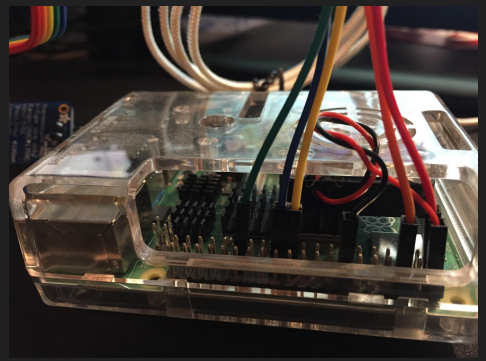
Hardware

- Raspberry Pi 4
- Adafruit MAX31855 Amplifier
- K-Type Thermocouple



Using a MAX31855 Thermocouple amplifier we are able to convert voltage readings into temperature readings

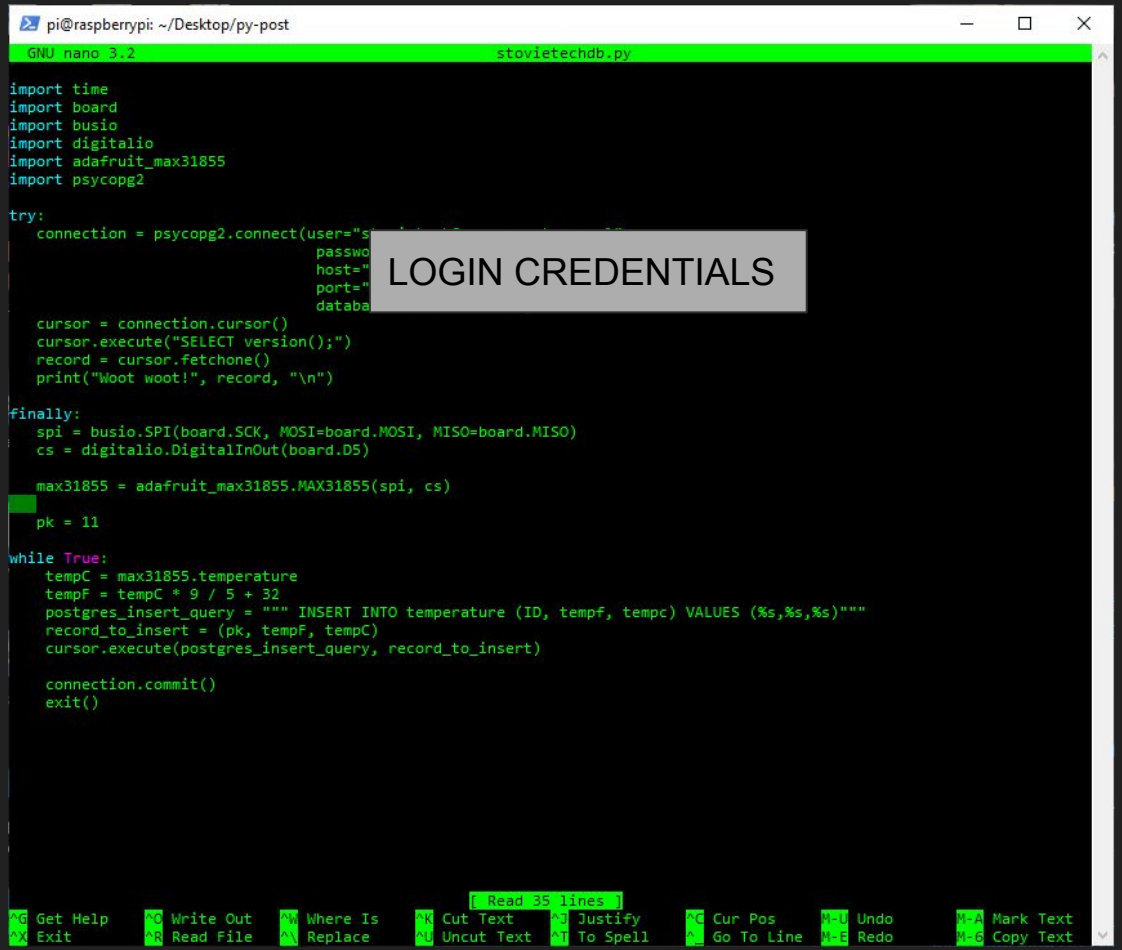




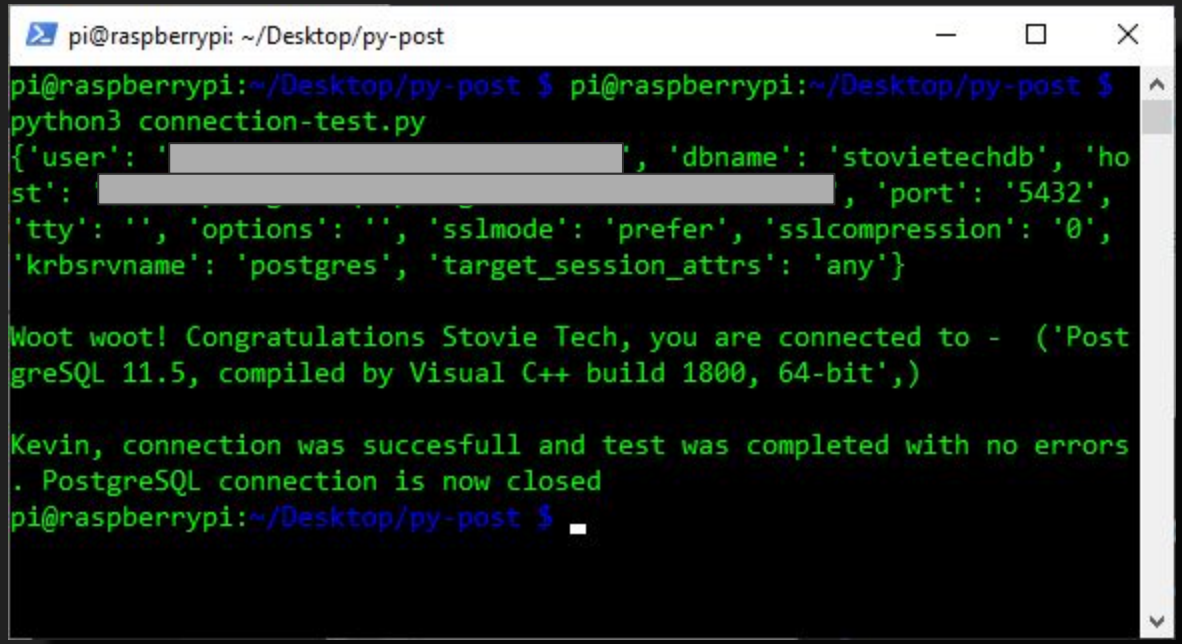
We connect the MAX31855 amplifier directly to the the Raspberry Pi 4 to access it’s SPI features.

Embedded Software

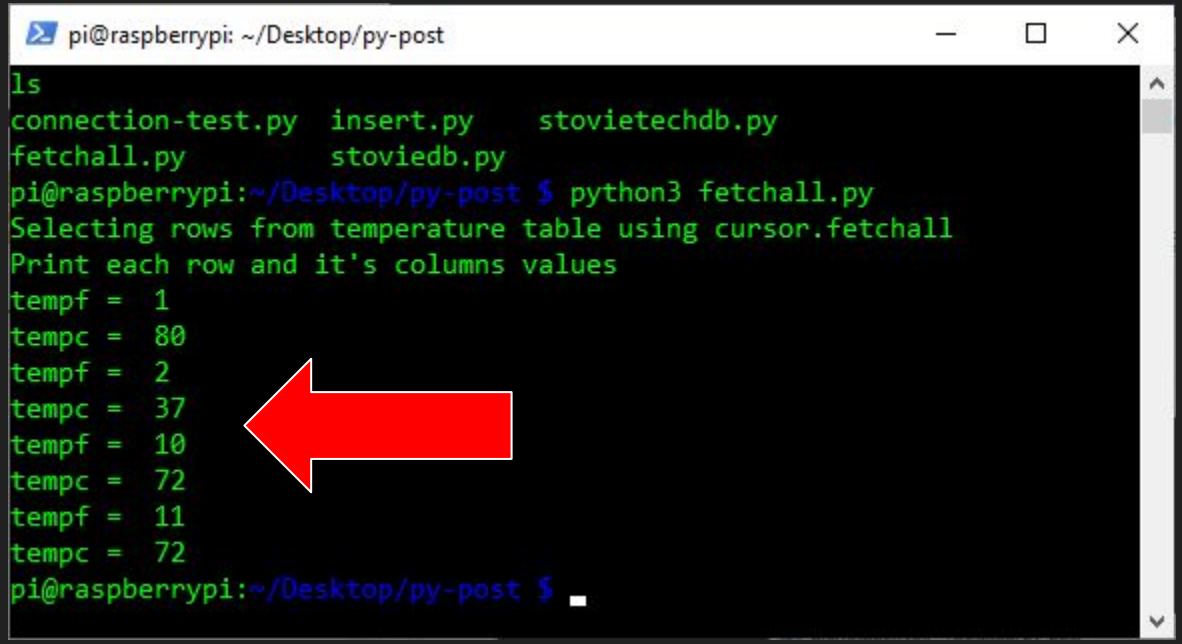
For our embedded software on the Raspberry Pi, we will utilize Python 3. Our Python program establishes a secure connection to a PostgreSQL DB hosted on Amazon RDS to insert temperature readings in an organized table.



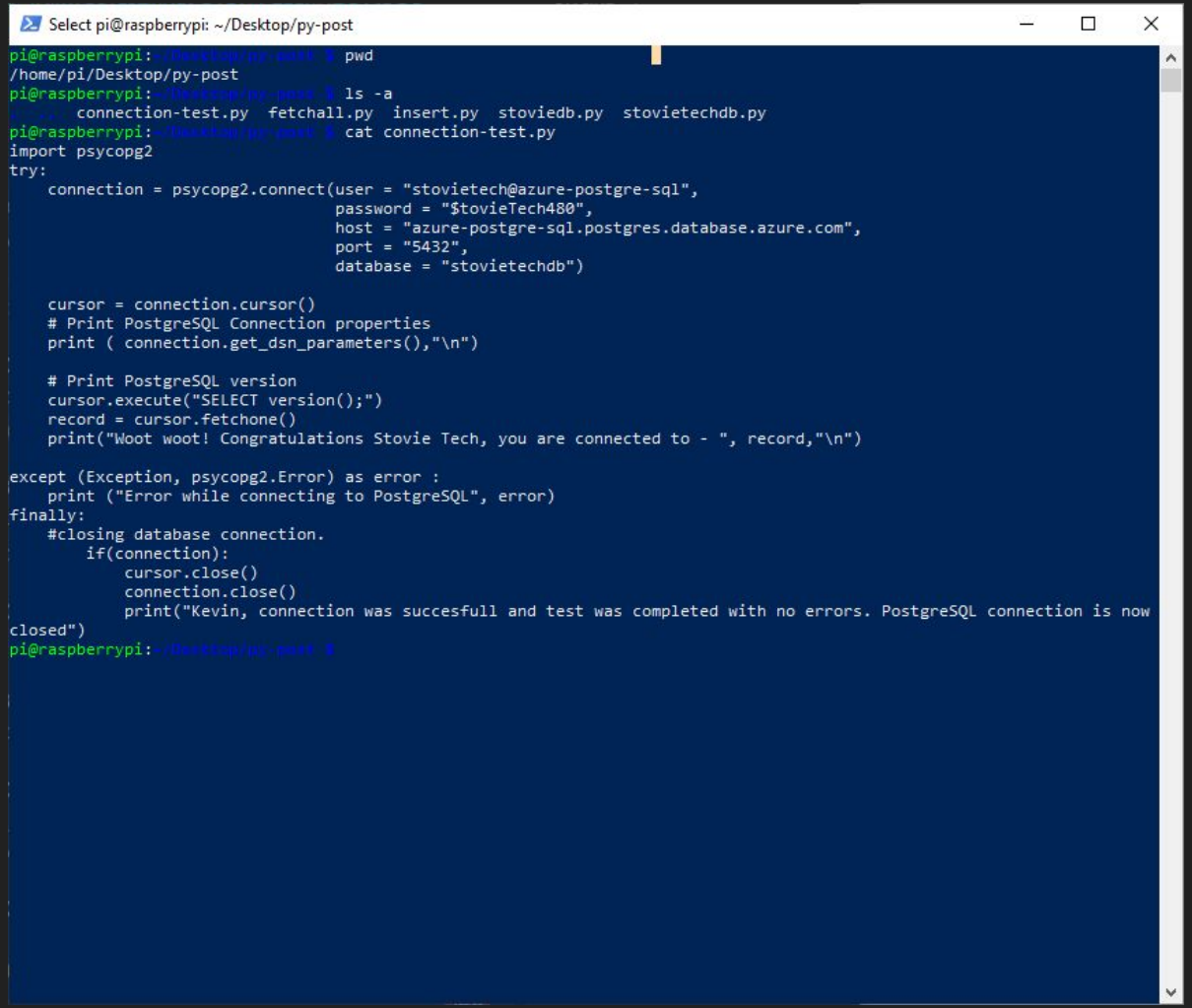
PostgreSQL Connection

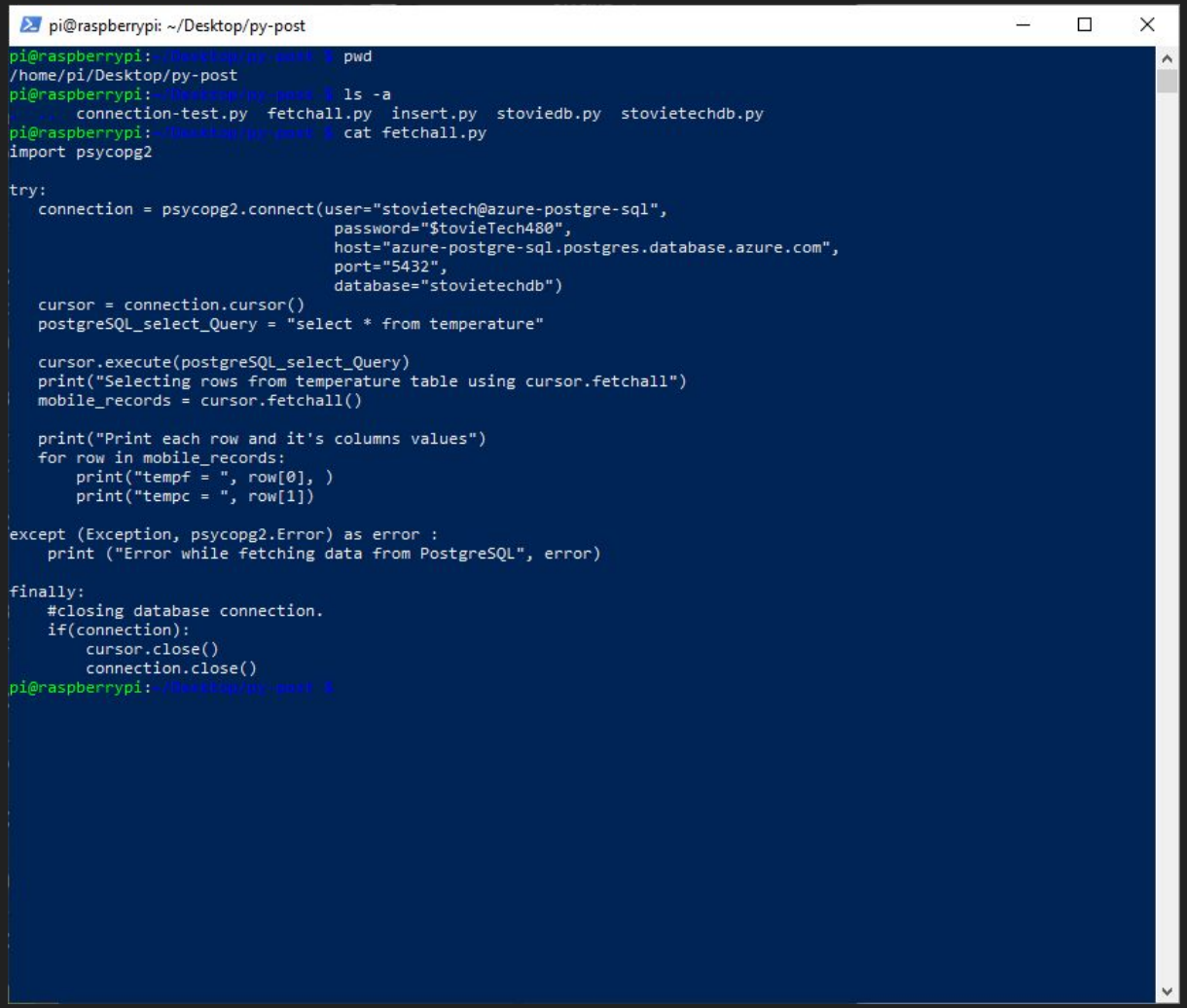


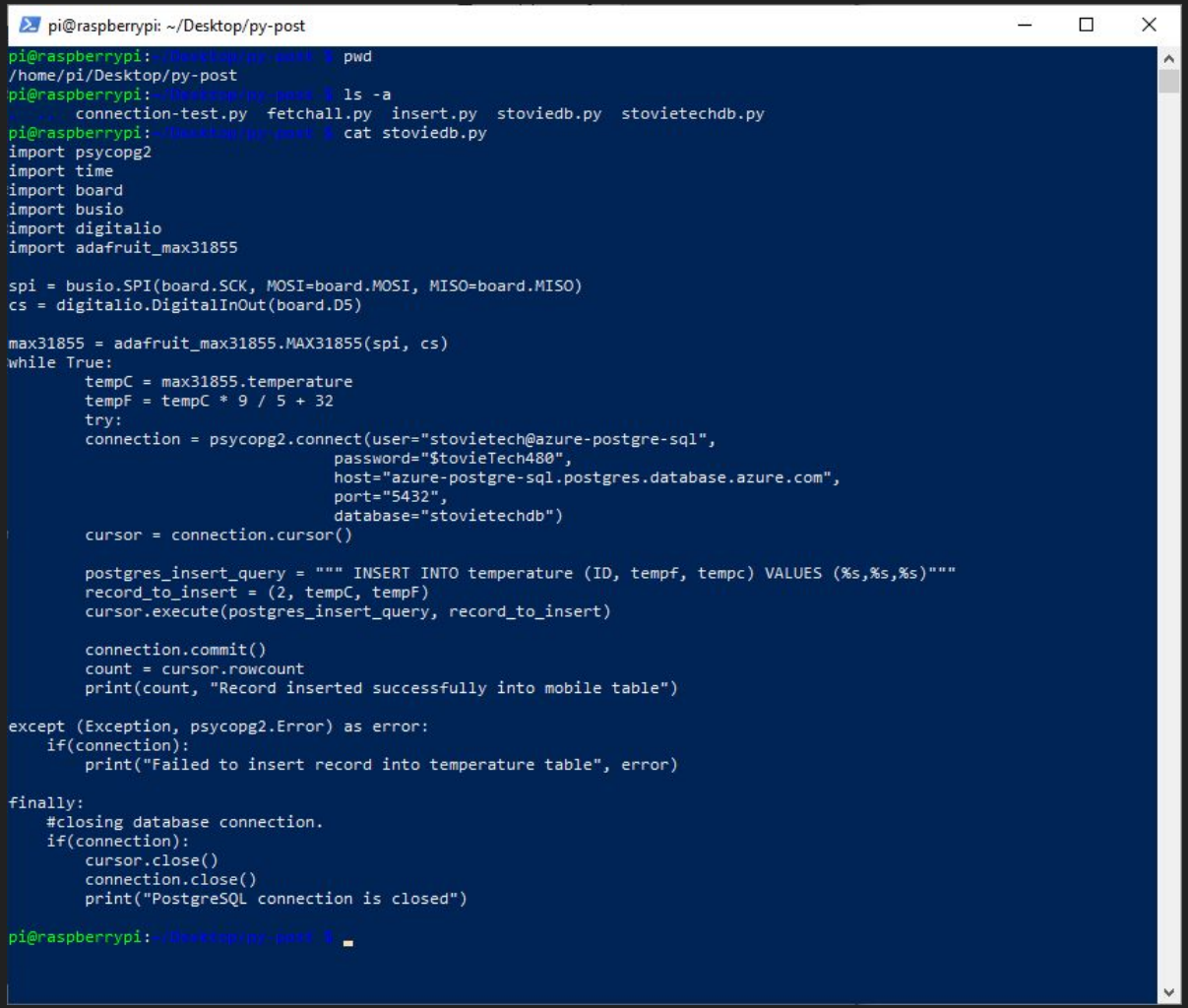
PostgreSQL Fetch Table Data



Python code used to
test connection to db, fetch from db, link sensor to db, and the actual sensor readings

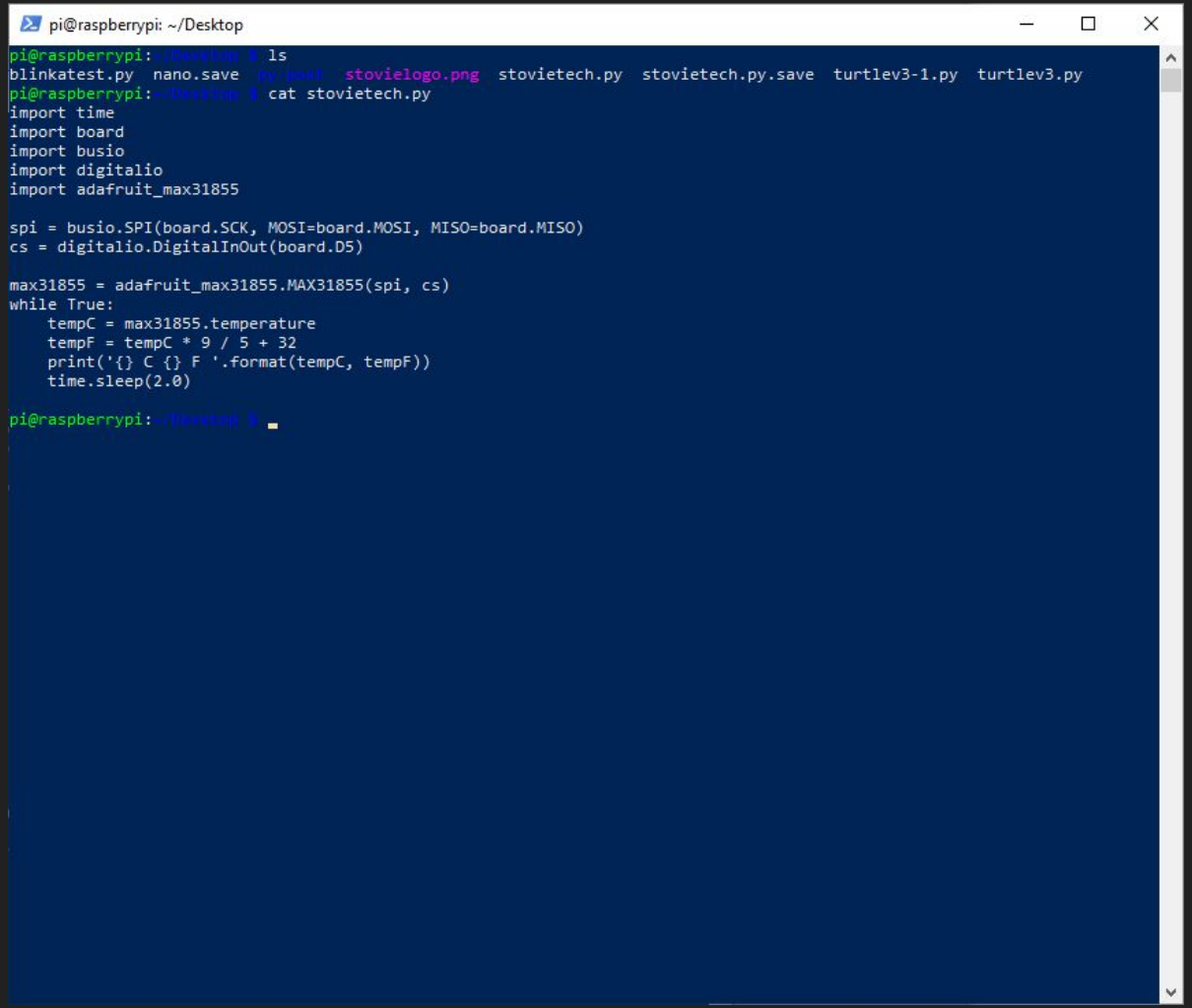

Select pi@raspberrypi: ~/Desktop/py-post
pi@raspberrypi:~/Desktop/py-post $ pwd
/home/pi/Desktop/py-post
pi@raspberrypi:~/Desktop/py-post $ ls -a
. .. connection-test.py fetchall.py insert.py stoviedb.py stovietechdb.py
pi@raspberrypi:~/Desktop/py-post $ cat connection-test.py
import psycopg2
try:
connection = psycopg2.connect(user = "stovietech@azure-postgre-sql",
password = "$tovieTech480",
host = "azure-postgre-sql.postgres.database.azure.com",
port = "5432",
database = "stovietechdb")
cursor = connection.cursor()
# Print PostgreSQL Connection properties
print (connection.get_dsn_parameters(),"\n")
# Print PostgreSQL version
cursor.execute("SELECT version();")
record = cursor.fetchone()
print("Woot woot! Congratulations Stovie Tech, you are connected to - ", record,"\n")
except (Exception, psycopg2.Error) as error :
print ("Error while connecting to PostgreSQL", error)
finally:
#closing database connection.
if(connection):
cursor.close()
connection.close()
print("Kevin, connection was succesfull and test was completed with no errors. PostgreSQL connection is now
closed")
pi@raspberrypi:~/Desktop/py-post $

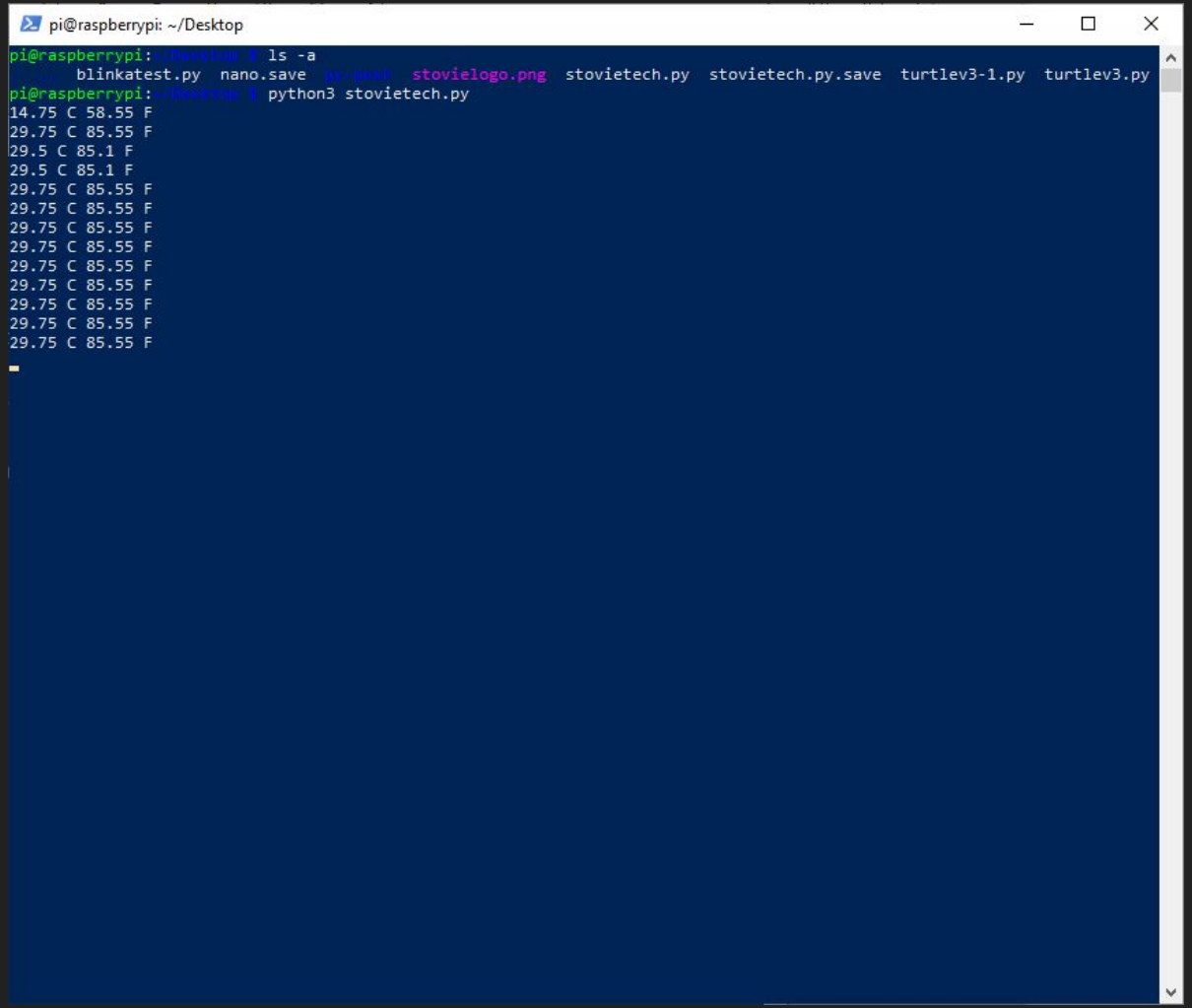

pi@raspberrypi: ~/Desktop/py-post
pi@raspberrypi:~/Desktop/py-post $ pwd
/home/pi/Desktop/py-post
pi@raspberrypi:~/Desktop/py-post $ ls -a
. .. connection-test.py fetchall.py insert.py stoviedb.py stovietechdb.py
pi@raspberrypi:~/Desktop/py-post $ cat fetchall.py
import psycopg2
try:
connection = psycopg2.connect(user="stovietech@azure-postgre-sql",
password="$tovieTech480",
host="azure-postgre-sql.postgres.database.azure.com",
port="5432",
database="stovietechdb")
cursor = connection.cursor()
postgreSQL_select_Query = "select * from temperature"
cursor.execute(postgreSQL_select_Query)
print("Selecting rows from temperature table using cursor.fetchall")
mobile_records = cursor.fetchall()
print("Print each row and it's columns values")
for row in mobile_records:
print("tempf = ", row[0],)
print("tempc = ", row[1])
except (Exception, psycopg2.Error) as error :
print ("Error while fetching data from PostgreSQL", error)
finally:
#closing database connection.
if(connection):
cursor.close()
connection.close()
pi@raspberrypi:~/Desktop/py-post $


pi@raspberrypi: ~/Desktop/py-post
pi@raspberrypi:~/Desktop/py-post $ pwd
/home/pi/Desktop/py-post
pi@raspberrypi:~/Desktop/py-post $ ls -a
. .. connection-test.py fetchall.py insert.py stoviedb.py stovietechdb.py
pi@raspberrypi:~/Desktop/py-post $ cat stoviedb.py
import psycopg2
import time
import board
import busio
import digitalio
import adafruit_max31855
spi = busio.SPI(board.SCK, MOSI=board.MOSI, MISO=board.MISO)
cs = digitalio.DigitalInOut(board.D5)
max31855 = adafruit_max31855.MAX31855(spi, cs)
while True:
tempC = max31855.temperature
tempF = tempC * 9 / 5 + 32
try:
connection = psycopg2.connect(user="stovietech@azure-postgre-sql",
password="$tovieTech480",
host="azure-postgre-sql.postgres.database.azure.com",
port="5432",
database="stovietechdb")
cursor = connection.cursor()
postgres_insert_query = """ INSERT INTO temperature (ID, tempf, tempc) VALUES (%s,%s,%s)"""
record_to_insert = (2, tempC, tempF)
cursor.execute(postgres_insert_query, record_to_insert)
connection.commit()
count = cursor.rowcount
print(count, "Record inserted successfully into mobile table")
except (Exception, psycopg2.Error) as error:
if(connection):
print("Failed to insert record into temperature table", error)
finally:
#closing database connection.
if(connection):
cursor.close()
connection.close()
print("PostgreSQL connection is closed")
pi@raspberrypi:~/Desktop/py-post $

pi@raspberrypi: ~/Desktop/py-post

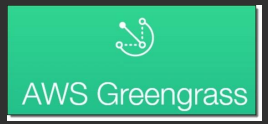
```
pi@raspberrypi:~/Desktop/py-post $ pwd
/home/pi/Desktop/py-post
pi@raspberrypi:~/Desktop/py-post $ ls -a
.  ..  connection-test.py  fetchall.py  insert.py  stoviedb.py  stovietechdb.py
pi@raspberrypi:~/Desktop/py-post $ catall stovietechdb.py
-bash: catall: command not found
pi@raspberrypi:~/Desktop/py-post $ cat stovietechdb.py
import time
import board
import busio
import digitalio
import adafruit_max31855
import psycopg2

try:
   connection = psycopg2.connect(user="stovietech@azure-postgre-sql",
                                 password="$tovieTech480",
                                 host="azure-postgre-sql.postgres.database.azure.com",
                                 port="5432",
                                 database="stovietechdb")
   cursor = connection.cursor()
   cursor.execute("SELECT version();")
   record = cursor.fetchone()
   print("Woot woot!", record, "\n")

finally:
   spi = busio.SPI(board.SCK, MOSI=board.MOSI, MISO=board.MISO)
   cs = digitalio.DigitalInOut(board.D5)

   max31855 = adafruit_max31855.MAX31855(spi, cs)

   pk = 11

while True:
    tempC = max31855.temperature
    tempF = tempC * 9 / 5 + 32
    postgres_insert_query = """ INSERT INTO temperature (ID, tempf, tempc) VALUES (%s,%s,%s)"""
    record_to_insert = (pk, tempF, tempC)
    cursor.execute(postgres_insert_query, record_to_insert)

    connection.commit()
    exit()
pi@raspberrypi:~/Desktop/py-post $
```


pi@raspberrypi: ~/Desktop
pi@raspberrypi:~/Desktop $ ls
blinkatest.py nano.save py.post stovielogo.png stovietech.py stovietech.py.save turtlev3-1.py turtlev3.py
pi@raspberrypi:~/Desktop $ cat stovietech.py
import time
import board
import busio
import digitalio
import adafruit_max31855
spi = busio.SPI(board.SCK, MOSI=board.MOSI, MISO=board.MISO)
cs = digitalio.DigitalInOut(board.D5)
max31855 = adafruit_max31855.MAX31855(spi, cs)
while True:
tempC = max31855.temperature
tempF = tempC * 9 / 5 + 32
print('{} C {} F '.format(tempC, tempF))
time.sleep(2.0)
pi@raspberrypi:~/Desktop $


pi@raspberrypi: ~/Desktop
pi@raspberrypi: ~/Desktop $ ls -a
. .. blinkatest.py nano.save __pycache__ stovielogo.png stovietech.py stovietech.py.save turtlev3-1.py turtlev3.py
pi@raspberrypi: ~/Desktop $ python3 stovietech.py
14.75 C 58.55 F
29.75 C 85.55 F
29.5 C 85.1 F
29.5 C 85.1 F
29.75 C 85.55 F
29.75 C 85.55 F
29.75 C 85.55 F
29.75 C 85.55 F
29.75 C 85.55 F
29.75 C 85.55 F
29.75 C 85.55 F
29.75 C 85.55 F
29.75 C 85.55 F

Amazon Web Services

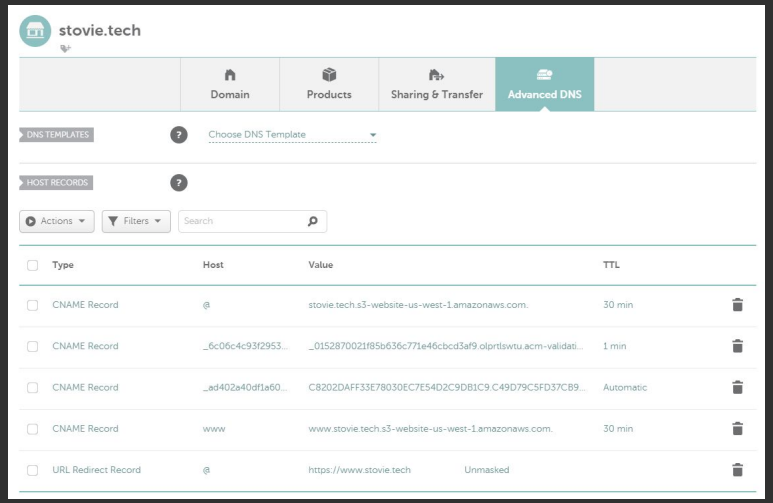
- AWS S3 bucket
- Certificate Manager
- Amazon RDS DB (Postgresql)
- GreenGrass IoT
- Amazon CloudFront



Domain Name Services

Stovie.tech
Obtained from NameCheap

- Set up to point to the AWS S3 bucket.
- SSL Certificate installed

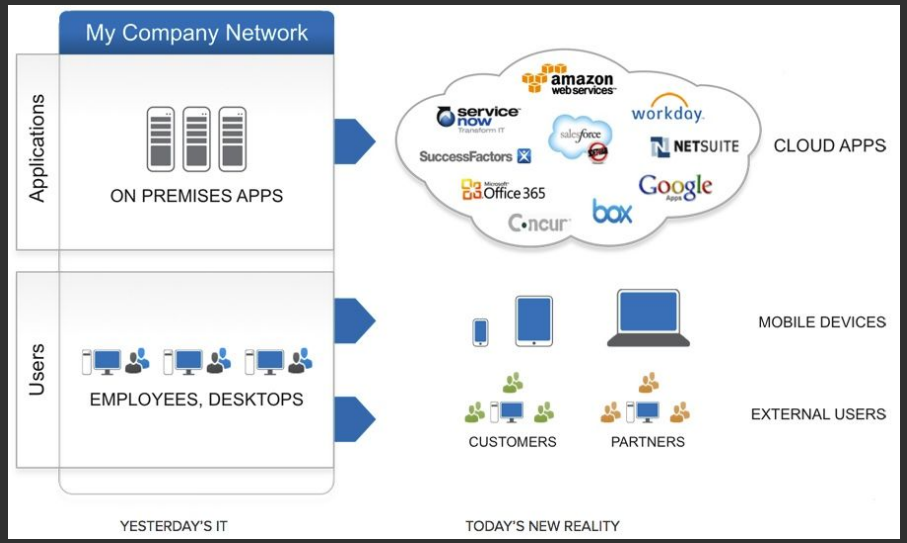


Okta

It's an enterprise-grade, identity management service, built for the cloud, but compatible with many on-premises applications.

Okta runs in the cloud, on a secure, reliable, extensively audited platform, which integrates deeply with on-premises applications, directories, and identity management systems.

The Okta solution was born of the unique challenges of how technology has grown and shifted in the growing diversity of devices, identity issues, security, employee mobility, vender partnership, and the exponential growth of unique application options.



Okta features

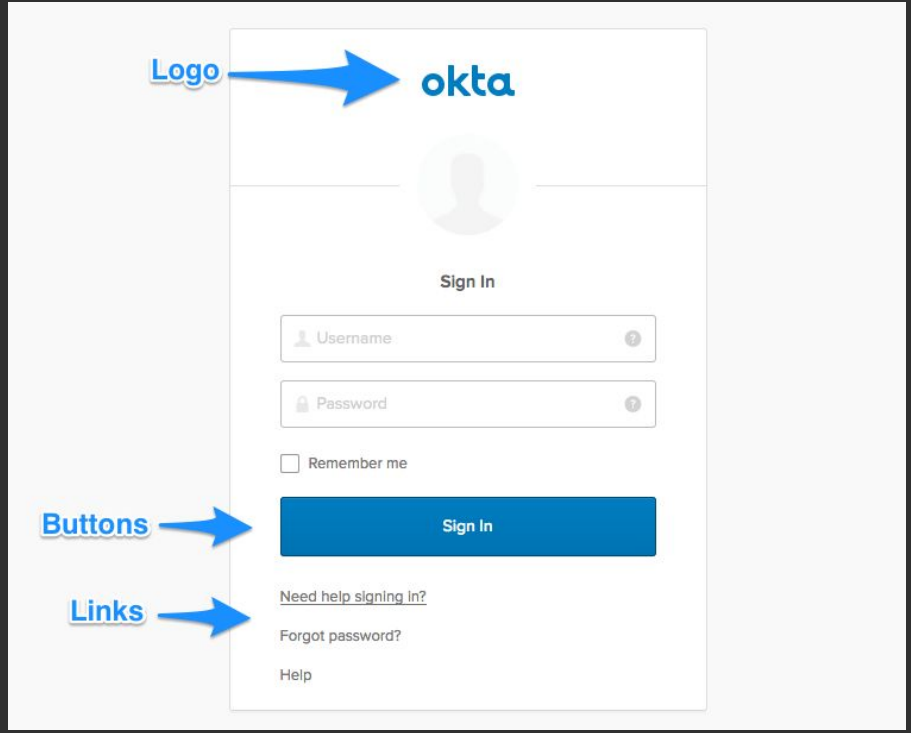
Okta features include Provisioning, Single Sign-On (SSO), Active Directory (AD) and LDAP integration, the centralized deprovisioning of users, multifactor authentication (MFA), mobile identity management, and flexible policies for organization security and control.

All of these functions are brought together through a network of pre-integrated applications called the Okta Integration Network (OIN). The OIN provides diverse integration options, enabling SSO login for every app your users need to access during their work day.

Okta login page

Okta will be the server that we use to host our login page for the application.

We connect to this using the OpenID connect client on the Okta Developer Dashboard.

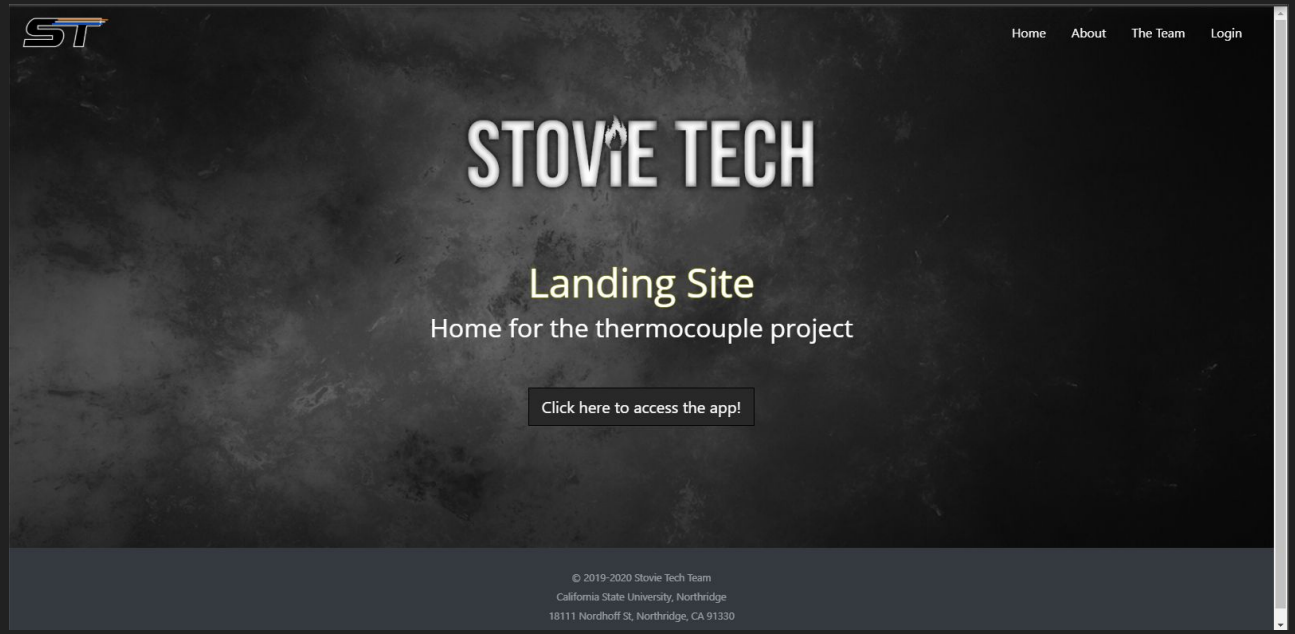


Website Development

- Implementations
  - HTML5, CSS, and Javascript
  - Bootstrap for UI Elements
  - Google Fonts
- Developed using Glitch and later on AWS Cloud 9 IDE.

index.html

- Designated as the main page
- Displays Name and Logo
- Shows navigations to
  - Log in to the IoT monitor app
  - About page
  - Team member page

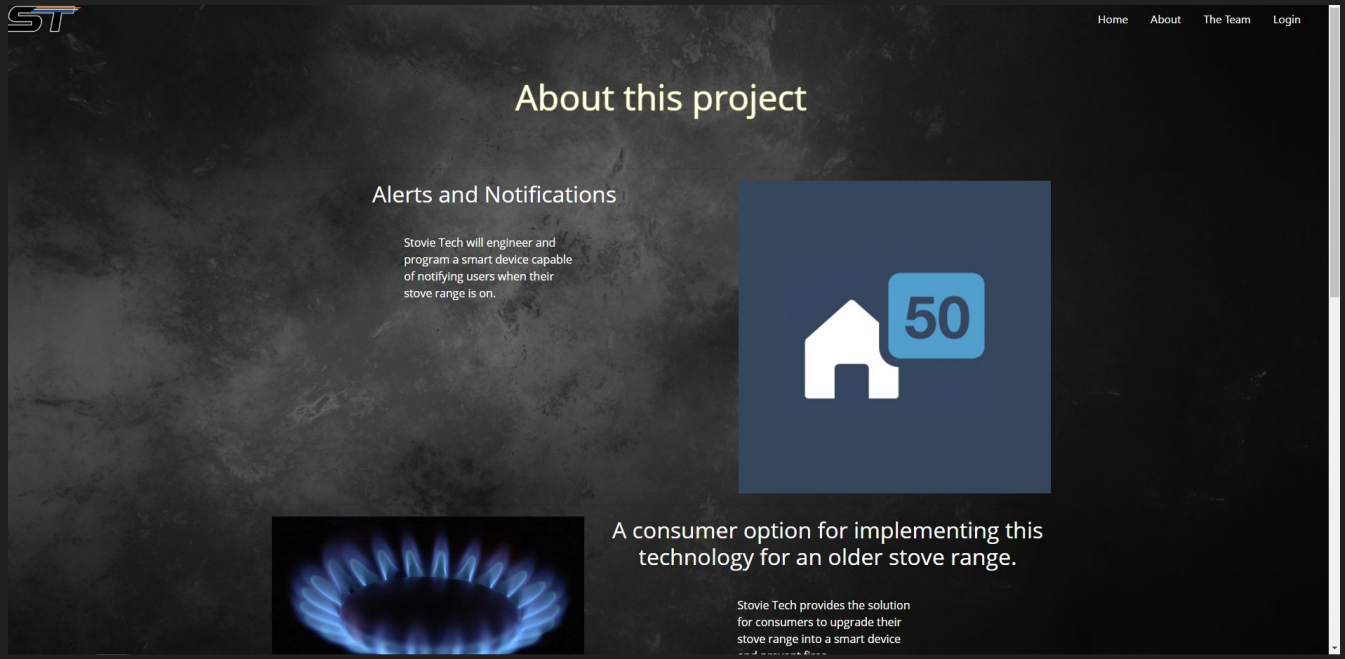


Code snippets for index.html

```
  <!-- below is code for the rest -->
  <header class="page-header header container-fluid">
    <div class="center">
      <img
        src="https://i.imgur.com/u3xO9Bq.png"
        class="picborder"
        href="#"
        height="250"
        width="500"
      />

      <h1 class="title center">Landing Site</h1>

      <h2 class="center">
        Home for the thermocouple project
      </h2>
    </div>

    <br />
    <br />

    <div class="center clickhere">
      <form action="login.html">
        <button class="btn btn-outline-secondary btn-lg" type="submit">
          Click here to access the app!
        </button>
      </form>
    </div>
  </header>

  <!-- <footer>
  <p class="footer">
    &copy 2019-2020 Stovie Tech Team
</p>

<p class="footer">
  California State University, Northridge
</p>

  <p class="footer">
  18111 Nordhoff St, Northridge, CA 91330
</p>
footer> -->

  <footer id="sticky-footer" class="py-4 bg-dark text-white-50">
    <div class="container text-center">
      <small>&copy 2019-2020 Stovie Tech Team</small>
      <br />
      <small>California State University, Northridge</small>
      <br />
      <small>18111 Nordhoff St, Northridge, CA 91330</small>
```

About Page

This page will be to inform users the purpose of our product and how it works.



About.html

```
<!--below is code for the rest-->

<main>
  <header>
    <br />
    <br />
    <h1 class="title center">
      About this project
    </h1>
    <br />
    <br />
  </header>
</main>

<div class="container marketing">
  <!-- START THE FEATURETTES -->

  <hr class="featurette-divider" />

  <div class="row featurette">
    <div class="col-md-7">
      <h2 class="featurette-heading">Alerts and Notifications</h2>
      <p>
        Stovie Tech will engineer and program a smart device capable of
        notifying users when their stove range is on.
      </p>
    </div>
    <div class="col-md-5">
      <img
        src="https://thumbs.gfycat.com/BoilingExaltedIaerismetalmark-small.gif"
        alt=""
        class="featurette-image img-fluid mx-auto"
      />
      <!--             <img src="https://cdn2.iconfinder.com/data/icons/mobile-apps-10/512/22_warning-512.png" alt="" class="featurette-image img-fluid mx-auto"> -->
    </div>
  </div>
  <hr class="featurette-divider" />

  <div class="row featurette">
    <div class="col-md-7 order-md-2">
      <h2 class="featurette-heading">
        A consumer option for implementing this technology for an older
        stove range.
      </h2>
      <p>
        Stovie Tech provides the solution for consumers to upgrade their
        stove range into a smart device and prevent fires.
      </p>
    </div>
    <div class="col-md-5 order-md-1">
      <!--             <img src="https://cdn4.iconfinder.com/data/icons/kitchen-1-1/512/36-512.png" alt="" class="featurette-image img-fluid mx-auto"> -->
      <img
```

Team Page

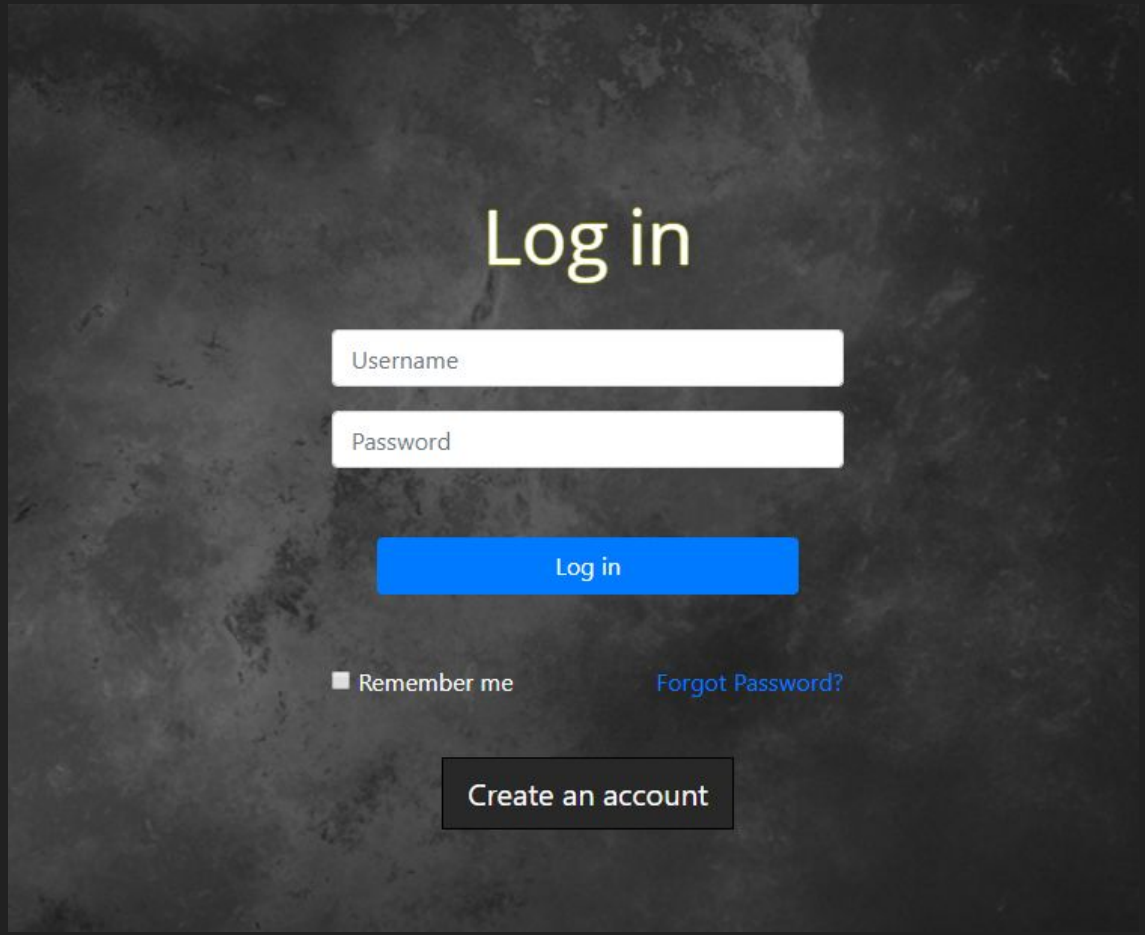
Displays the names of all the developers to this projects and their roles.

Team Page

```
  <title>Stovie Team</title>
</head>
<body>
  <!-- below is code for the nav bar -->
  <nav class="navbar navbar-expand-md">
    <!--   <a class="navbar-brand" href="#">Team Name</a> -->
    <a href="index.html"
      ><img
        class="navbar-brand"
        src="https://i.imgur.com/mQi0qOi.png"
        height="50"
        width="110"
    /></a>
    <button
      class="navbar-toggler navbar-dark"
      type="button"
      data-toggle="collapse"
      data-target="#main-navigation"
    >
      <span class="navbar-toggler-icon"></span>
    </button>
    <div class="collapse navbar-collapse" id="main-navigation">
      <ul class="navbar-nav">
        <li class="nav-item">
          <a class="nav-link" href="index.html">Home</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="about.html">About</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="team.html">The Team</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="login.html">Login</a>
        </li>
      </ul>
    </div>
  </nav>
  <!-- code to list all members and descriptions -->

  <div class="bg-black py-5">
    <div class="container py-5">
      <h1 class="title center">The Stovie Tech Team</h1>

      <div class="row text-center">
        <!-- Team item-->
        <div class="col-xl-3 col-sm-6 mb-5">
          <div class="rounded shadow-sm py-5 px-4">
            <img
              src="https://i.imgur.com/o5Gw9ek.jpg"
              alt=""
```

Login Page

For users to log in to their respective accounts to access the dashboard of the IoT monitor. Also links to password reset should password be forgotten, and account creation for new users.

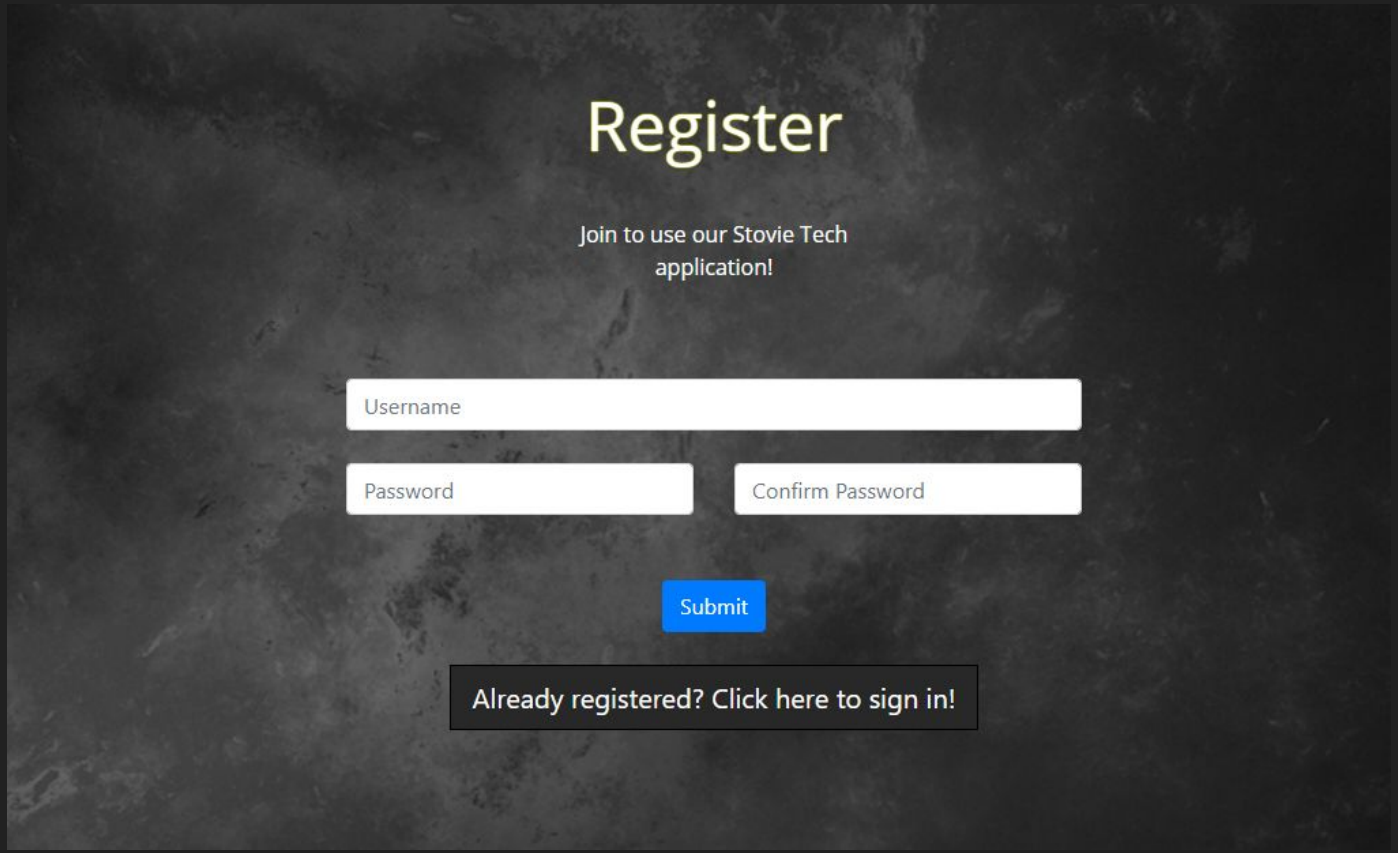


Login Page

```
  </head>
  <body>
    <br />
    <br />
    <style type="text/css"></style>
    <div class="login-form">
<!--        <form action="/examples/actions/confirmation.php" method="post"> -->
        <h1 class="title center">Log in</h1>
        <br />
        <div class="form-group">
          <input
            type="text"
            class="form-control"
            placeholder="Username"
            required="required"
          />
        </div>
        <div class="form-group">
          <input
            type="password"
            class="form-control"
            placeholder="Password"
            required="required"
          />
        </div>
        <div class="form-group">
          <form action="dashboard.html">
          <button type="submit" class="btn btn-primary btn-block">
            Log in
          </button>
          </form>
        </div>
        <div class="clearfix loginwhite">
          <label class="pull-left checkbox-inline"
            ><input type="checkbox" /> Remember me</label
          >
          <a href="forgot-password.html" class="pull-right">Forgot Password?</a>
        </div>
<!--        </form> -->
      <div class="center clickhere">
        <!-- Change URL to good URL -->
        <form action="signup.html">
          <button class="btn btn-outline-secondary btn-lg" type="submit">
            Create an account
          </button>
        </form>
      </div>
    </div>

    <footer id="sticky-footer" class="py-4 bg-dark text-white-50">
      <div class="container text-center">
        <small>&copy 2019-2020 Stovie Tech Team</small>
```

Sign Up Page

New users will sign up here with a username and password.



Sign up page

```
<!-- signup form -->

<section>
  <div class="container">
    <div class="row justify-content-center">
      <div class="col-12 col-md-8 col-lg-8 col-xl-6">
        <div class="row">
          <div class="col text-center">
            <h1 class="title">Register</h1>
            <p class="text-h3">Join to use our Stovie Tech application!</p>
          </div>
        </div>
        <!--           <div class="row align-items-center">
        <div class="col mt-4">
          <input type="text" class="form-control" placeholder="Company Name">
        </div>
      </div> -->
        <div class="row align-items-center mt-4">
          <div class="col">
            <input
              type="username"
              class="form-control"
              placeholder="Username"
            />
          </div>
        </div>
        <div class="row align-items-center mt-4">
          <div class="col">
            <input
              type="password"
              class="form-control"
              placeholder="Password"
            />
          </div>
          <div class="col">
            <input
              type="password"
              class="form-control"
              placeholder="Confirm Password"
            />
          </div>
        </div>
        <div class="row justify-content-start mt-4">
          <div class="col center">
            <!--               <div class="form-check">
            <label class="form-check-label">
              <input type="checkbox" class="form-check-input">
              I Read and Accept <a href="https://www.froala.com">Terms and Conditions</a>
            </label>
          </div> -->

            <button class="btn btn-primary mt-4">Submit</button>
```

Forgot password

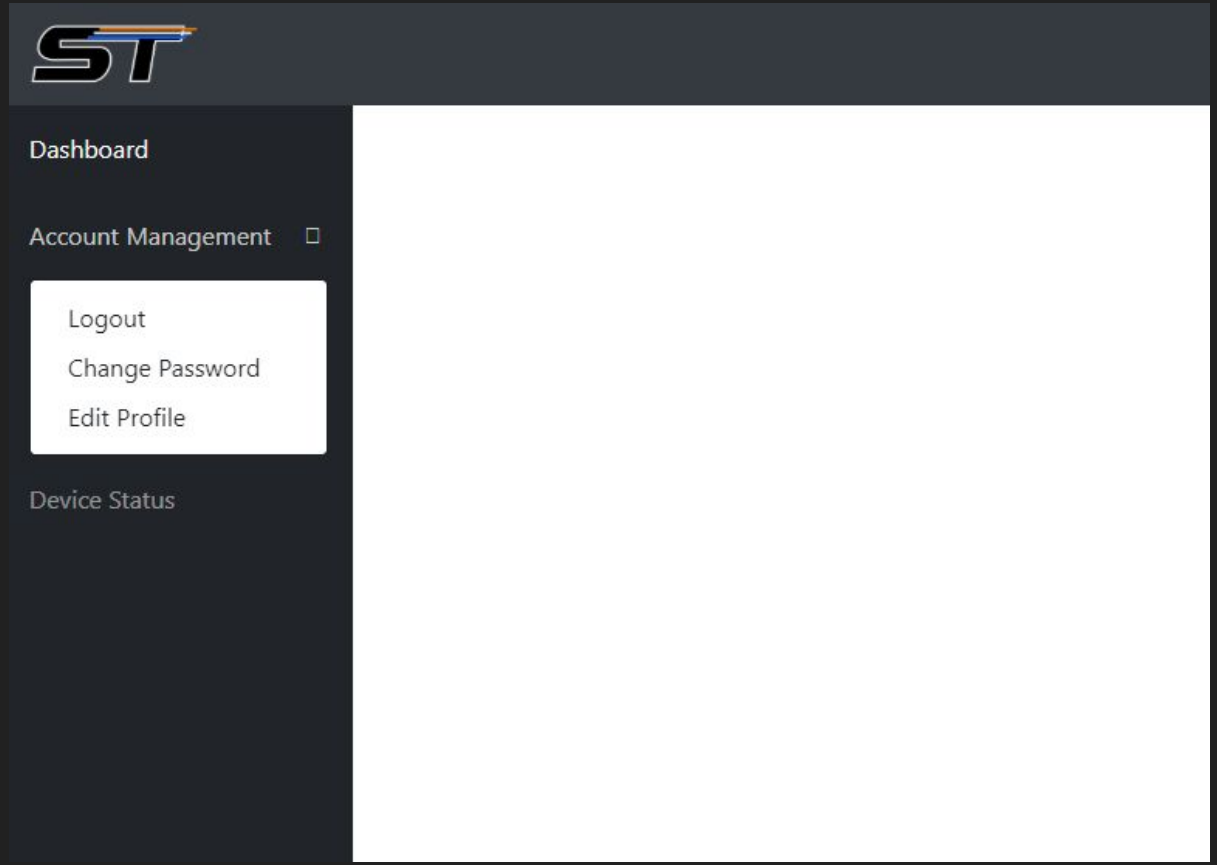
```
<!DOCTYPE html>
<html lang="en">

<head>

  <meta charset="utf-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
  <meta name="description" content="">
  <meta name="author" content="">

  <title>SB Admin - Forgot Password</title>

  <!-- Custom fonts for this template-->
  <link href="vendor/fontawesome-free/css/all.min.css" rel="stylesheet" type="text/css">

  <!-- Custom styles for this template-->
  <link href="css/sb-admin.css" rel="stylesheet">

</head>

<body class="bg-dark">

  <div class="container">
    <div class="card card-login mx-auto mt-5">
      <div class="card-header">Reset Password</div>
      <div class="card-body">
        <div class="text-center mb-4">
          <h4>Forgot your password?</h4>
        </div>
        <form>
          <div class="form-group">
            <div class="form-label-group">
              <input type="email" id="inputEmail" class="form-control" placeholder="Enter email address" required="required" autofocus="autofocus">
              <label for="inputuser">Enter username</label>
            </div>
          </div>
          <a class="btn btn-primary btn-block" href="login.html">Reset Password</a>
        </form>
        <div class="text-center">
          <a class="d-block small mt-3" href="signup.html">Register an Account</a>
          <a class="d-block small" href="login.html">Login Page</a>
        </div>
      </div>
    </div>
  </div>

  <!-- Bootstrap core JavaScript-->
  <script src="vendor/jquery/jquery.min.js"></script>
  <script src="vendor/bootstrap/js/bootstrap.bundle.min.js"></script>
```

Change Password

```
<!-- signup form -->

<section>
  <div class="container">
    <div class="row justify-content-center">
      <div class="col-12 col-md-8 col-lg-8 col-xl-6">
        <div class="row">
          <div class="col text-center">
            <h1 class="title">Change Password</h1>
            <p class="text-h3">Follow form below!</p>
          </div>
        </div>
        <!--          <div class="row align-items-center">
        <div class="col mt-4">
          <input type="text" class="form-control" placeholder="Company Name">
        </div>
      </div> -->
        <div class="row align-items-center mt-4">
          <div class="col">
            <input
              type="username"
              class="form-control"
              placeholder="Username"
            />
          </div>
        </div>
        <div class="row align-items-center mt-4">
          <div class="col">
            <input
              type="password"
              class="form-control"
              placeholder="Old Password"
            />
          </div>
        </div>
        <div class="row align-items-center mt-4">
          <div class="col">
            <input
              type="password"
              class="form-control"
              placeholder="New Password"
            />
          </div>
        </div>
        <div class="row align-items-center mt-4">
          <div class="col">
            <input
              type="password"
              class="form-control"
              placeholder="Confirm New Password"
            />
          </div>
```

Dashboard

Once users sign in they will be given the dashboard to monitor device status and manage their accounts. More to be added later.

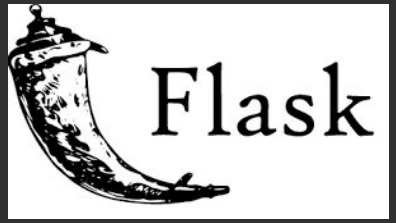


Dashboard page

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta
      name="viewport"
      content="width=device-width, initial-scale=1, shrink-to-fit=no"
    />
    <meta name="description" content="" />
    <meta name="author" content="" />

    <title>Stovie Tech Dashboard</title>

    <!-- Custom fonts for this template-->
    <link
      href="vendor/fontawesome-free/css/all.min.css"
      rel="stylesheet"
      type="text/css"
    />

    <!-- Page level plugin CSS-->
    <link href="vendor/datatables/dataTables.bootstrap4.css" rel="stylesheet" />

    <!-- Custom styles for this template-->
    <link href="css/sb-admin.css" rel="stylesheet" />
  </head>

  <body id="page-top">
    <nav class="navbar navbar-expand navbar-dark bg-dark static-top">
      <!--     <a class="navbar-brand mr-1" href="index.html">Stovie Tech (Testing)</a> -->
      <a href="index.html"
        ><img
          class="navbar-brand"
          src="https://i.imgur.com/mQi0qOi.png"
          height="50"
          width="110"
      /></a>

      <button
        class="btn btn-link btn-sm text-white order-1 order-sm-0"
        id="sidebarToggle"
        href="#"
      >
        <i class="fas fa-bars"></i>
      </button>

      <!-- Navbar Search -->
      <!--     <form class="d-none d-md-inline-block form-inline ml-auto mr-0 mr-md-3 my-2 my-md-0">
      <div class="input-group">
        <input type="text" class="form-control" placeholder="Search for..." aria-label="Search" aria-describedby="basic-addon2">
        <div class="input-group-append">
```

Back-end Development & Deployment

- Core Technologies
  - Python
  - Flask
  - AWS
  - Sqlalchemy
- The backend is developed with Flask & SQLAlchemy deployed on the AWS.



main.py

index.html

about.html

team.html

auth.py

register.html

login.html

logout.html

dashboard.html

stovie.py

monitor.db

Project Structure

```
- stovie
---- templates
-------- base.html <!-- contains common layout and links -->
-------- index.html <!-- show the home page -->
-------- login.html <!-- show the login form -->
-------- dashboard.html <!-- show the profile page -->
-------- signup.html <!-- show the signup form -->
---- __init__.py <!-- setup our app -->
---- auth.py <!-- the auth routes for our app -->
---- main.py <!-- the non-auth routes for our app -->
---- models.py <!-- our user model -->
```

Flask Training 〉 main.py

flask

Proj...

Flask Training ~/Flask
- static
- templates
- main.py
- routingExercise.py

External Libraries

Scratches and Consoles

base.html | main.py | index.html | signup.html | 404.html | thankyou.html

```
from flask import Flask, render_template, request

app = Flask(__name__)


@app.route('/')
def index():
    return render_template('index.html')


@app.route('/signup')
def signup():
    return render_template('signup.html')


@app.route('/thankyou')
def thankyou():
    firstName = request.args.get('firstName')
    lastName = request.args.get('lastName')

    return render_template('thankyou.html', firstName=firstName, lastName=lastName)


@app.errorhandler(404)
def pageNotFound(e):
    return render_template('404.html'), 404


if __name__ == '__main__':
    app.run(debug=True)
```

Terminal: Local

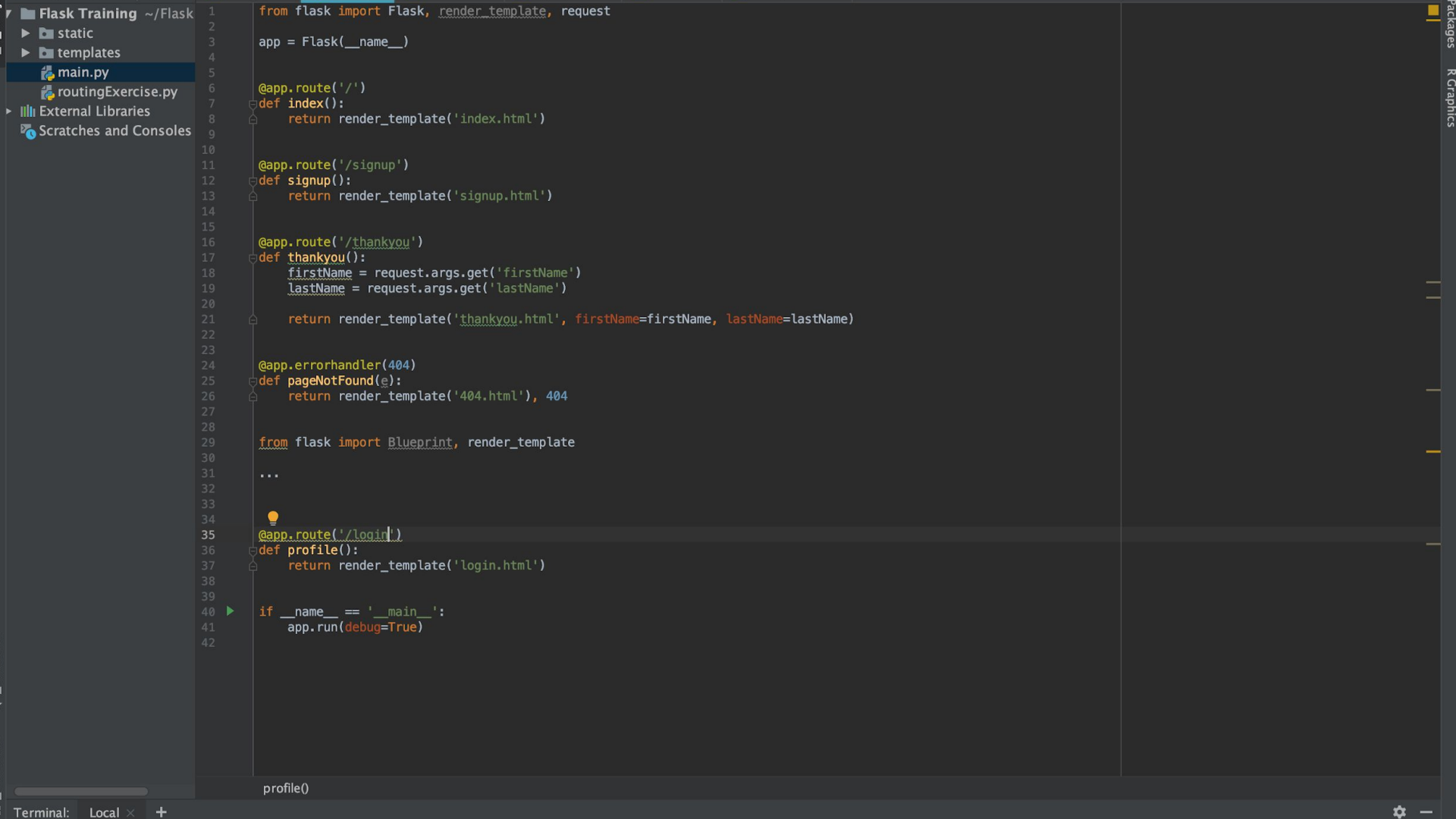
```
 * Debugger PIN: 246-740-913
127.0.0.1 - - [24/Nov/2019 21:10:23] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [24/Nov/2019 21:10:25] "GET /asd HTTP/1.1" 404 -
 * Detected change in '/Users/mirza/Flask Training/app.py', reloading
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 246-740-913
 * Detected change in '/Users/mirza/Flask Training/app.py', reloading
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 246-740-913
127.0.0.1 - - [24/Nov/2019 21:11:33] "GET /rolo HTTP/1.1" 404 -
```

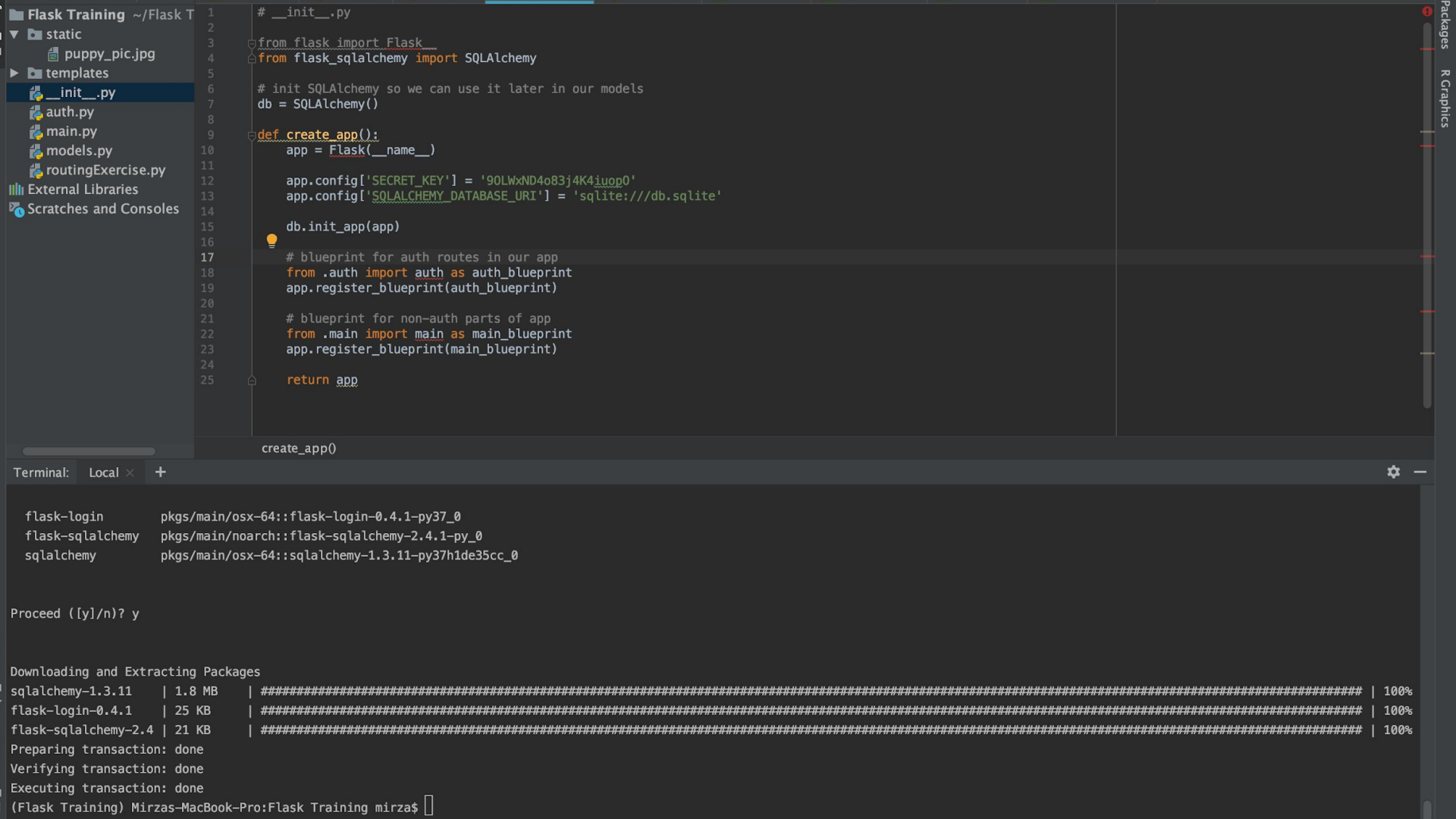
Python Console | Terminal | 4: Run | 6: TODO | 3 Event Log

IDE and Plugin Updates: PyCharm is ready to update. (yesterday 18:56)

1:1 LF UTF-8 4 spaces Python 3.7 (Flask Training)

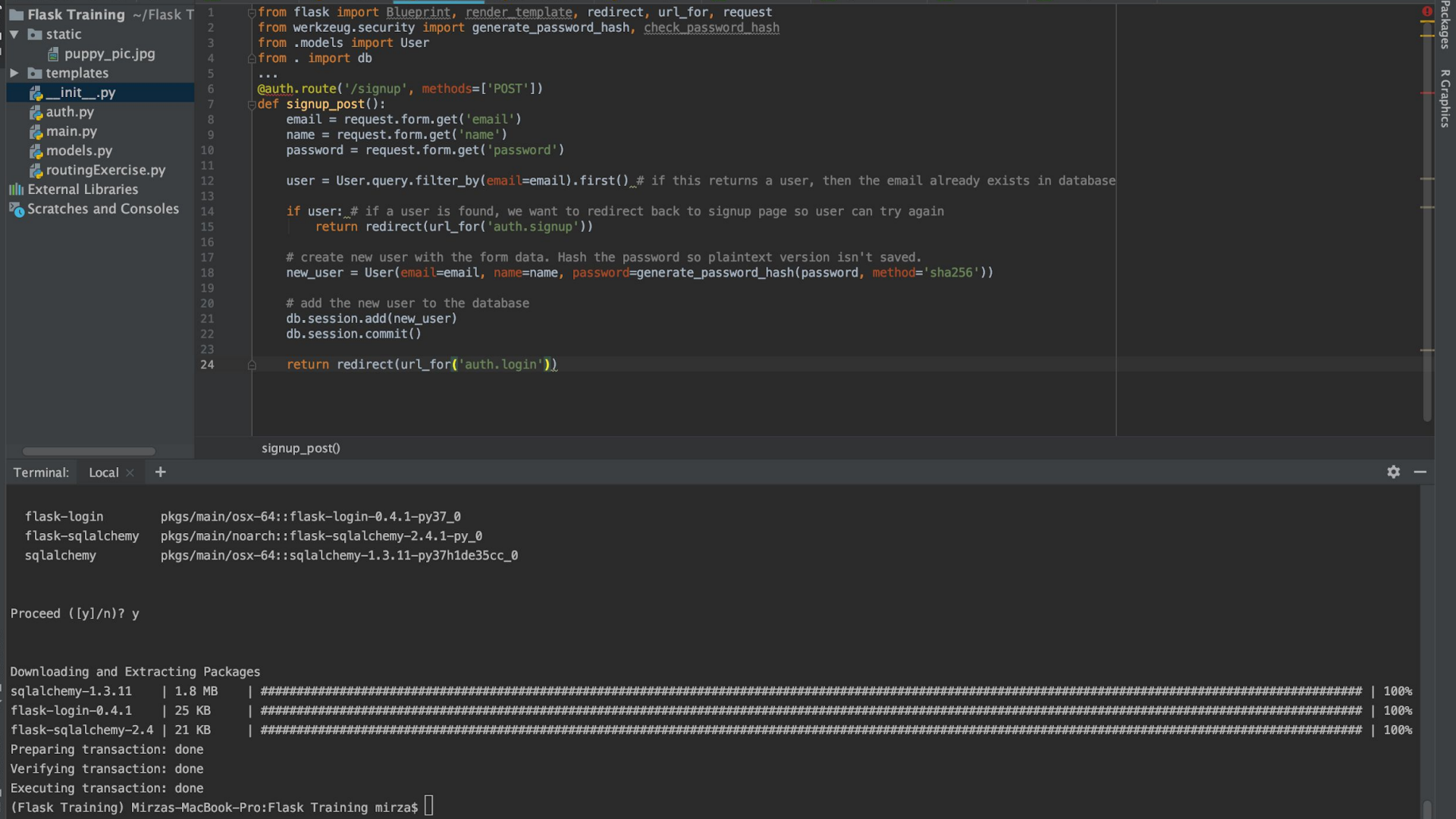
R Packages R Graphics


Flask Training ~/Flask
static
templates
main.py
routingExercise.py
External Libraries
Scratches and Consoles
from flask import Flask, render_template, request
app = Flask(__name__)
@app.route('/')
def index():
return render_template('index.html')
@app.route('/signup')
def signup():
return render_template('signup.html')
@app.route('/thankyou')
def thankyou():
firstName = request.args.get('firstName')
lastName = request.args.get('lastName')
return render_template('thankyou.html', firstName=firstName, lastName=lastName)
@app.errorhandler(404)
def pageNotFound(e):
return render_template('404.html'), 404
from flask import Blueprint, render_template
...
@app.route('/login')
def profile():
return render_template('login.html')
if __name__ == '__main__':
app.run(debug=True)
profile()
Terminal:
Local


Flask Training ~/Flask T
static
puppy_pic.jpg
templates
__init__.py
auth.py
main.py
models.py
routingExercise.py
External Libraries
Scratches and Consoles
# __init__.py
from flask import Flask
from flask_sqlalchemy import SQLAlchemy
# init SQLAlchemy so we can use it later in our models
db = SQLAlchemy()
def create_app():
app = Flask(__name__)
app.config['SECRET_KEY'] = '9OLWxND4o83j4K4iuopO'
app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///db.sqlite'
db.init_app(app)
# blueprint for auth routes in our app
from .auth import auth as auth_blueprint
app.register_blueprint(auth_blueprint)
# blueprint for non-auth parts of app
from .main import main as main_blueprint
app.register_blueprint(main_blueprint)
return app
create_app()
Terminal: Local
flask-login pkgs/main/osx-64::flask-login-0.4.1-py37_0
flask-sqlalchemy pkgs/main/noarch::flask-sqlalchemy-2.4.1-py_0
sqlalchemy pkgs/main/osx-64::sqlalchemy-1.3.11-py37h1de35cc_0
Proceed ([y]/n)? y
Downloading and Extracting Packages
sqlalchemy-1.3.11 | 1.8 MB | ########## | 100%
flask-login-0.4.1 | 25 KB | ########## | 100%
flask-sqlalchemy-2.4 | 21 KB | ########## | 100%
Preparing transaction: done
Verifying transaction: done
Executing transaction: done
(Flask Training) Mirzas-MacBook-Pro:Flask Training mirza$

Password Hashing

- For user authentication, Stovie will need to store their usernames and passwords.
- For security purposes, we will never want to store the raw text of the password.
- We are using a hash function to help with this.
- A hash function is an algorithm that can take in a document (password) and return back a secure hash digest.


Flask Training ~/Flask T
static
puppy_pic.jpg
templates
__init__.py
auth.py
main.py
models.py
routingExercise.py
External Libraries
Scratches and Consoles
from flask import Blueprint, render_template, redirect, url_for, request
from werkzeug.security import generate_password_hash, check_password_hash
from .models import User
from . import db
...
@auth.route('/signup', methods=['POST'])
def signup_post():
email = request.form.get('email')
name = request.form.get('name')
password = request.form.get('password')
user = User.query.filter_by(email=email).first() # if this returns a user, then the email already exists in database
if user: # if a user is found, we want to redirect back to signup page so user can try again
return redirect(url_for('auth.signup'))
# create new user with the form data. Hash the password so plaintext version isn't saved.
new_user = User(email=email, name=name, password=generate_password_hash(password, method='sha256'))
# add the new user to the database
db.session.add(new_user)
db.session.commit()
return redirect(url_for('auth.login'))
signup_post()
Terminal:
Local
flask-login pkgs/main/osx-64::flask-login-0.4.1-py37_0
flask-sqlalchemy pkgs/main/noarch::flask-sqlalchemy-2.4.1-py_0
sqlalchemy pkgs/main/osx-64::sqlalchemy-1.3.11-py37h1de35cc_0
Proceed ([y]/n)? y
Downloading and Extracting Packages
sqlalchemy-1.3.11 | 1.8 MB | ########## | 100%
flask-login-0.4.1 | 25 KB | ########## | 100%
flask-sqlalchemy-2.4 | 21 KB | ########## | 100%
Preparing transaction: done
Verifying transaction: done
Executing transaction: done
(Flask Training) Mirzas-MacBook-Pro:Flask Training mirza$
Packages
R Graphics

Project Scheduling with Trello