Slide 7

Slide 6

Slide 5

Slide 4

Slide 3

Slide 2

Slide 1

Aliyev Omar
Haider Zaheer
Michael Hernandez
Nephthro Pierre

http://hawkincloud.github.io/dev/

Slide 7

Slide 6

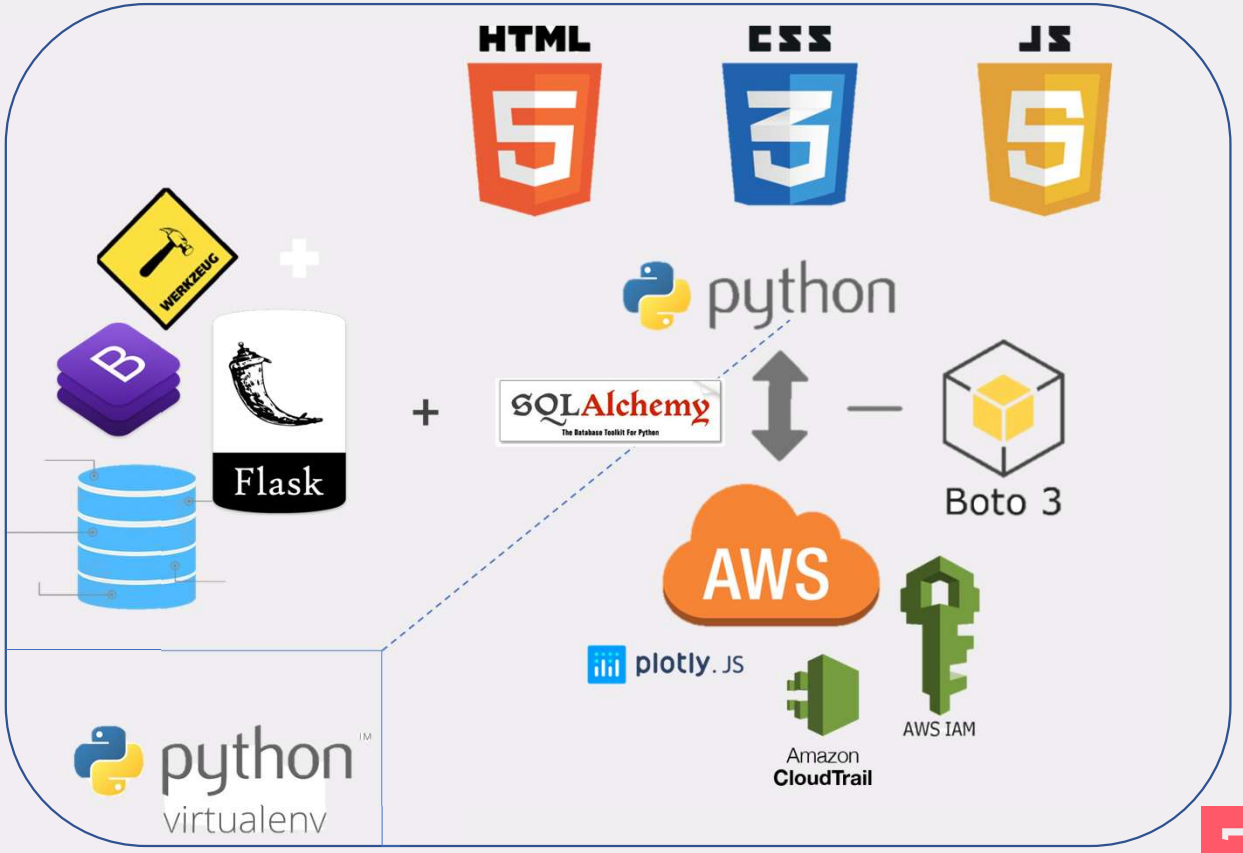
Slide 5

Slide 4

Slide 3

Slide 2

Code Infrastructure



Slide 1

Slide 7

Slide 6

Slide 5

Slide 4

Slide 3

Dependencies & pip list
OS Environment

```
Babel==2.11.0
boto3==1.26.79
botocore==1.29.79
click==8.1.3
colorama==0.4.6
contourpy==1.0.7
cycler==0.11.0
dnspython==2.3.0
dominate==2.7.0
email-validator==1.3.1
Flask==2.2.3
Flask-Bootstrap==3.3.7.1
Flask-Charts==1.7
Flask-Login==0.6.2
Flask-Moment==1.0.5
Flask-SQLAlchemy==3.0.3
Flask-WTF==1.1.1
fonttools==4.38.0
greenlet==2.0.2
idna==3.4
itsdangerous==2.1.2
Jinja2==3.1.2
jmespath==1.0.1
jsonify==0.5
kiwisolver==1.4.4
MarkupSafe==2.1.2
matplotlib==3.7.0
numpy==1.24.2
packaging==23.0
Pillow==9.4.0
pip-check==2.8.1
prettytable==3.5.0
pyparsing==3.0.9
python-dateutil==2.8.2
pytz==2022.7.1
s3transfer==0.6.0
six==1.16.0
SQLAlchemy==2.0.4
terminaltables==3.1.10
typing_extensions==4.5.0
urllib3==1.26.13
visitor==0.1.3
wcwidth==0.2.5
Werkzeug==2.2.3
WTForms==3.0.1
youtube-dl==2021.12.17
```

```
pip install virtualenv
```

```
from flask import Flask, jsonify, render_template, redirect, url_for
from flask_bootstrap import Bootstrap
import os
from flask_wtf import FlaskForm
from wtforms import StringField, PasswordField, BooleanField
from wtforms.validators import InputRequired, Length, Email
import email_validator
from flask_sqlalchemy import SQLAlchemy
from werkzeug.security import generate_password_hash, check_password_hash
from flask_login import LoginManager, UserMixin, login_user, login_required, logout_user, current_user
import boto3
```


```
pip freeze > requirements.txt
pip install -r requirements.txt
```

```
.
└── HawkinCloud/
    └── venv/
        ├── Source/
        │   ├── __pycahce__
        │   ├── instance
        │   ├── static/
        │   │   ├── pics
        │   │   ├── dashboard.css
        │   │   ├── signin.css
        │   │   └── starter-template.css
        │   ├── templates/
        │   │   ├── charts.html
        │   │   ├── dashboard.html
        │   │   ├── index.html
        │   │   ├── login.html
        │   │   ├── sg_graph.html
        │   │   └── signup.html
        │   ├── app.py
        │   └── requriements.txt
        ├── Include
        ├── Lib/
        │   └── Site-packages/
        │       └── ...........
        ├── Scripts/
        │   ├── activate
        │   ├── activate.bat
        │   └── jp.py
        └── pyvenv.cfg
```

Slide 2

Slide 1

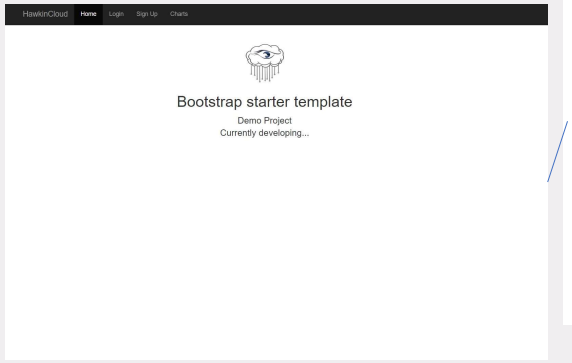
Slide 7

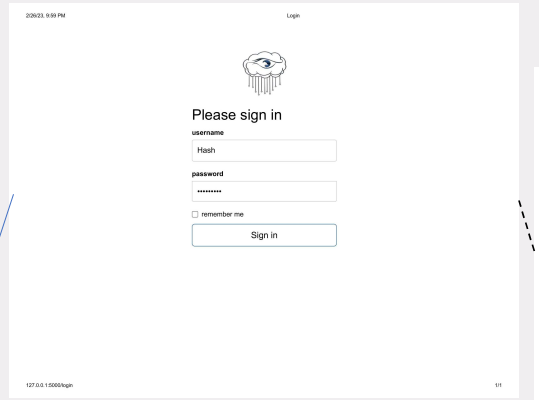
Slide 6

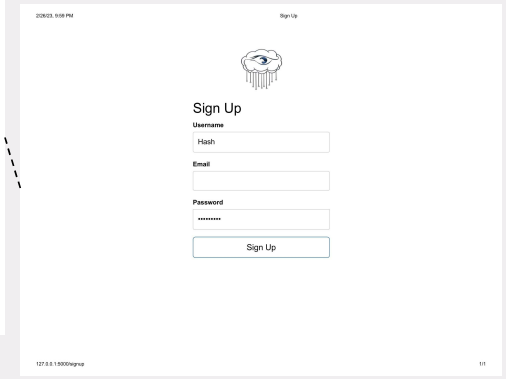
Slide 5

Slide 4

UI Interface & Database Flask

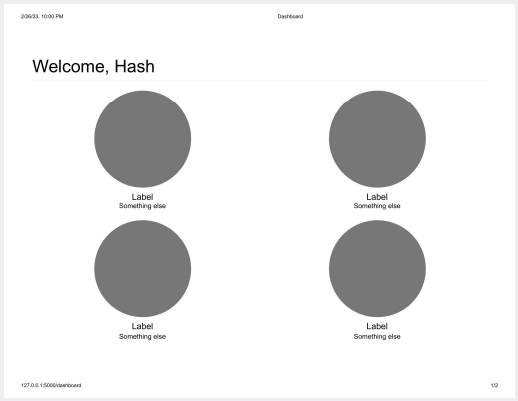







```
app.config['SECRET_KEY'] = "[redacted]"
app.config['SQLALCHEMY_DATABASE_URI'] = [redacted]database.db'
Bootstrap(app)
db = SQLAlchemy(app)
login_manager = LoginManager()
login_manager.init_app(app)
login_manager.login_view = 'login'
```

```
@app.route('/signup', methods=['GET', 'POST'])
def signup():
    form = RegisterForm()
    if form.validate_on_submit():
        # return '<h1>' + form.username.data + ' ' + form.email.data + ' ' + form.password.data + ' </
        hashed_password = generate_password_hash(form.password.data, method='sha256')
        new_user = User(username=form.username.data, email=form.email.data, password=hashed_password)
        db.session.add(new_user)
        db.session.commit()
```




```
gitaliyev@Aliyev:[redacted]$ sqlite3 database.db
SQLite version 3.37.2 2022-01-06 13:25:41
Enter ".help" for usage hints.
sqlite> .tables
user
sqlite> select * from user;
1|Hash|hash@hash.com|sha256$snBYTOqRzjL371VY$ef940b4795c54e0447c90b9a6122ca8f19780453247f09cfc43541fe745e8a7c
sqlite>
```

Slide 3

Slide 2

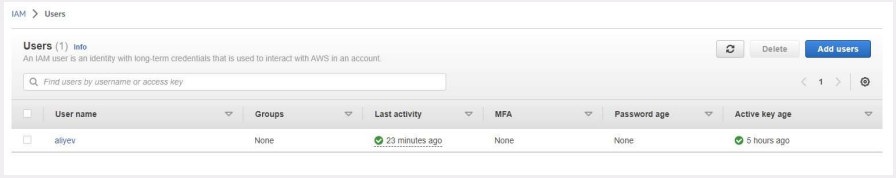
Slide 1

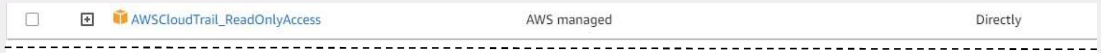
Slide 7

Slide 6

Slide 5

AWS CLI, IAM Configure & Boto3 Visualize Data






```
PowerShell 7.3.3
PS C:\Users\Aliyev> aws sts get-caller-identity
{
    "UserId": [redacted]
    "Account": [redacted]
    "Arn": [redacted]:user/aliyev"
}

PS C:\Users\Aliyev> cd .\.aws\
PS C:\Users\Aliyev\.aws> ls

    Directory: C:\Users\Aliyev\.aws

Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a---          2/26/2023  5:04 PM             46 config
-a---          2/26/2023  5:04 PM            119 credentials
```

```
@app.route('/sg_data')
def sg_data():
    # Connect to AWS and get the data
    client = boto3.client('cloudtrail')
    # Parse the CloudTrail response
    security_groups_created = 0
    security_groups_deleted = 0
    security_groups_modified = 0

    events = ['CreateSecurityGroup', 'DeleteSecurityGroup', 'AuthorizeSecurityGroupIngress',
              'AuthorizeSecurityGroupEgress', 'RevokeSecurityGroupIngress', 'RevokeSecurityGroupEgress']
    # Construct the CloudTrail query

    for evnt in events:
        query = {
            'LookupAttributes': [
                {
                    'AttributeKey': 'EventName',
                    'AttributeValue': evnt
                }
            ],
            'MaxResults': 50
        }
        # Run the CloudTrail query
        response = client.lookup_events(**query)

        if evnt == 'CreateSecurityGroup':
            security_groups_created = len(response['Events'])
        elif evnt == 'DeleteSecurityGroup':
            security_groups_deleted = len(response['Events'])
        else:
            security_groups_modified += len(response['Events'])

    data = {'sg_create_count': security_groups_created,
            'sg_delete_count': security_groups_deleted,
            'sg_modify_count': security_groups_modified}
    return jsonify(data)
```

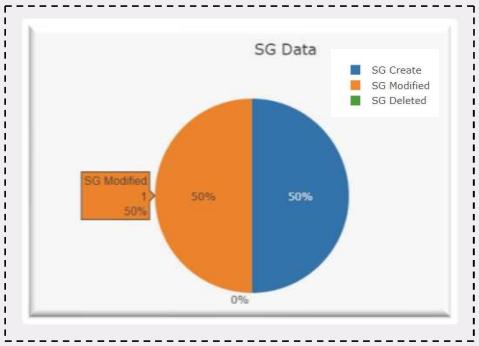
```
<!-- Include Plotly.js -->
    <script src="https://cdn.plot.ly/plotly-latest.min.js"></script>

{% endblock %}

{% block content %}


<body>
    <div id="pie-chart"></div>
    <script>
        function updatePieChart() {
            // Make a request to the Flask route to get the data
            fetch('/sg_data')
                .then(response => response.json())
                .then(data => {
                    // Create the pie chart using Plotly.js
                    var trace1 = {
                        values: [data.sg_create_count, data.sg_delete_count, data.sg_modify_count],
                        labels: ['SG Create', 'SG Deleted', 'SG Modified'],
                        type: 'pie'
                    };
                    var layout = {
                        title: 'SG Data',
                        height: 400
                    };
                    var data = [trace1];
                    Plotly.newPlot('pie-chart', data, layout);
                });
        }

        // Call the updatePieChart function every 30 seconds
        setInterval(updatePieChart, 10000);
    </script>
```

Slide 4

Slide 3

```
127.0.0.1 - - [26/Feb/2023 22:35:42] "GET /charts HTTP/1.1" 200 -
127.0.0.1 - - [26/Feb/2023 22:35:56] "GET /sg_data HTTP/1.1" 200 -
127.0.0.1 - - [26/Feb/2023 22:36:05] "GET /sg_data HTTP/1.1" 200 -
127.0.0.1 - - [26/Feb/2023 22:36:15] "GET /sg_data HTTP/1.1" 200 -
```

Output



Slide 2

Slide 1

Slide 7

Slide 6

Management

Brief Status Overview

HawkinCloud Status

Since the most recent day of class—presentation day, our group has made substantial efforts in organizing and distributing workload amongst one another, respectively.

With just over 1 month into the initiation of the AWS project, HawkinCloud, we have focused on workload distribution in the form of deliverables assigned to one another, respectively.

As we progress, we are meeting more than once a week, either virtually or in person. Our means of communication have been predominantly on Discord server and over Zoom

Slide 5

Slide 4

Slide 3

Slide 2

Slide 1

Where We Are & Where We're Going

(As of the end of February 2023)

Slide 6
Slide 5
Slide 4
Slide 3
Slide 2
Slide 1

Conclusion

We decided as a group to put our main focus into learning and adapting with AWS from Amazon. With our basic structure and understanding with our initial project goal, we will continue to further develop our front-end and back-end and clear up any hurdles we face along the way to create a functional site and code that can be fixed towards each deadline.

Slide 7
Slide 6
Slide 5
Slide 4
Slide 3
Slide 2
Slide 1

Thanks for
Listening with
great patience

Slide 7
Slide 6
Slide 5
Slide 4
Slide 3
Slide 2
Slide 1