FLASK

By Examples

Hello World

```
|____ app
|   |____ __init__.py
|   |____ routes.py
```


MSU DENVER

Hello World

```
export FLASK_APP=app
```

app > __init__.py > ...

```python
from flask import Flask

app = Flask("My Web App")

from app import routes
```

app > routes.py > ...

```python
from app import app

@app.route('/')
def index():
    return "Hello, World!"
```


MSU DENVER

SQLAlchemy

```
|____ app
|   |____ __init__.py
|   |____ routes.py
|   |____ models.py
|   |____ templates
|   |   |____ students.html
```


MSU DENVER

SQLAlchemy

app > models.py > ...

```python
from app import db

class Student(db.Model):
    __tablename__ = 'students'
    id = db.Column(db.String, primary_key=True)
    name = db.Column(db.String)
```


MSU DENVER

app > __init__.py > ...

```python
from flask import Flask

app = Flask("My Web App")

# db initialization
from flask_sqlalchemy import SQLAlchemy
db = SQLAlchemy()
app.config["SQLALCHEMY_DATABASE_URI"] = "sqlite:///app.db"
db.init_app(app)

# db from models
from app import models
with app.app_context():
    db.create_all()

from app import routes
```


MSU DENVER

app > routes.py > ...

```python
from app import app, db
from flask import render_template, redirect, url_for, request
from app.models import Student

@app.route('/students')
def list_students():
    students = Student.query.all()
    return render_template("students.html", students=students)

@app.route('/students/create', methods=['GET','POST'])
def create_employee():
    id = request.args.get('id')
    name = request.args.get('name')
    student = Student(id=id, name=name)
    db.session.add(student)
    db.session.commit()
    return redirect(url_for('list_students'))
```


MSU DENVER

SQLAlchemy

templates > <> students.html > ...

```
1    {% for student in students %}
2        <p>{{ student.id }} - {{ student.name }}</p>
3    {% endfor %}
```


MSU DENVER

SQLAlchemy + Form

```
|___ app
|  |___ __init__.py
|  |___ routes.py
|  |___ models.py
|  |___ forms.py
|  |___ templates
|  |  |___ students.html
|  |  |___ students_create.html
```


MSU DENVER

app > __init__.py > ...

```python
from flask import Flask

app = Flask("My Web App")
app.config['SECRET_KEY'] = 'you-will-never-guess'

# db initialization
from flask_sqlalchemy import SQLAlchemy
db = SQLAlchemy()
app.config["SQLALCHEMY_DATABASE_URI"] = "sqlite:///app.db"
db.init_app(app)

# db from models
from app import models
with app.app_context():
    db.create_all()

from app import routes
```


MSU DENVER

SQLAlchemy + Forms

app > forms.py > ...

```python
from flask_wtf import FlaskForm
from wtforms import StringField, SubmitField, validators
from wtforms.validators import DataRequired

class StudentsCreateForm(FlaskForm):
    id = StringField('Id', validators=[DataRequired()])
    name = StringField('Name', validators=[DataRequired()])
    submit = SubmitField('Submit')
```


MSU DENVER

SQLAlchemy + Forms

app > routes.py > create_student

```python
from app import app, db
from flask import render_template, redirect, url_for, request
from app.models import Student
from app.forms import StudentsCreateForm

@app.route('/students')
def list_students():
    students = Student.query.all()
    return render_template("students.html", students=students)
```


MSU DENVER

SQLAlchemy + Forms

```
11  @app.route('/students/create', methods=['GET','POST'])
12  def create_student():
13      form = StudentsCreateForm()
14      if form.validate_on_submit():
15          student = Student(id=form.id.data, name=form.name.data)
16          db.session.add(student)
17          db.session.commit()
18          return redirect(url_for('list_students'))
19      else:
20          return render_template('students_create.html', form=form)
```


MSU DENVER

SQLAlchemy + Forms

templates > students_create.html > ...

```
1    <form action="" method="POST" novalidate>
2        {{ form.hidden_tag() }}
3        <p>{{ form.id.label }} <br> {{ form.id(size=5) }}</p>
4        <p>{{ form.name.label }} <br> {{ form.name(size=30) }}</p>
5        <p>{{ form.submit() }}</p>
6    </form>
```


MSU DENVER

Flask-Login

- Flask-Login provides user session management for Flask
- Stores the active user's ID in the Flask Session, and let you easily log them in and out

MSU DENVER

Flask-Login

- If an app's secret key is set, you can use sessions in Flask applications
- A session makes it possible to remember information from one request to another by using a signed cookie

MSU DENVER

Flask-Login

```
|____ app
|   |____ __init__.py
|   |____ routes.py
|   |____ models.py
|   |____ forms.py
|   |____ templates
|   |   |____ users.html
|   |   |____ login.html
|   |   |____ login_failed.html
```


MSU DENVER

Flask-Login

app > models.py > ...

```python
from app import db
from flask_login import UserMixin

class User(db.Model, UserMixin):
    __tablename__ = 'users'
    id = db.Column(db.String, primary_key=True)
    password = db.Column(db.String)
```


MSU DENVER

Flask-Login

app > forms.py > ...

```
1   from flask_wtf import FlaskForm
2   from wtforms import StringField, PasswordField, SubmitField, validators
3   from wtforms.validators import DataRequired
4
5   class LoginForm(FlaskForm):
6       id = StringField('Username', validators=[DataRequired()])
7       password = PasswordField('Password', validators=[DataRequired()])
8       submit = SubmitField('Submit')
```


MSU DENVER

app > __init__.py > load_user

```python
from flask import Flask

app = Flask("My Web App")
app.config['SECRET_KEY'] = 'you-will-never-guess'

# db initialization
from flask_sqlalchemy import SQLAlchemy
db = SQLAlchemy()
app.config["SQLALCHEMY_DATABASE_URI"] = "sqlite:///app.db"
db.init_app(app)

# db from models
from app import models
with app.app_context():
    db.create_all()
```


MSU DENVER

Flask-Login

```
17    # login manager
18    from flask_login import LoginManager
19    login_manager = LoginManager()
20    login_manager.init_app(app)
21
22    from app.models import User
23
24    # user_loader callback
25    @login_manager.user_loader
26    def load_user(id):
27        return db.session.query(User).filter(User.id==id).one()
28
29    from app import routes
```


MSU DENVER

Flask-Login

app > routes.py > create_user

```python
from app import app, db
from flask import render_template, redirect, url_for, request
from app.models import User
from app.forms import LoginForm
from flask_login import login_user, login_required, logout_user

@app.route('/users')
@login_required
def list_users():
    users = User.query.all()
    return render_template("users.html", users=users)
```


MSU DENVER

Flask-Login

```
13  @app.route('/users/create', methods=['GET','POST'])
14  def create_user():
15      form = LoginForm()
16      if form.validate_on_submit():
17          user = User(id=form.id.data, password=form.password.data)
18          db.session.add(user)
19          db.session.commit()
20          return redirect(url_for('login'))
21      else:
22          return render_template('login.html', form=form)
```


MSU DENVER

```
24  @app.route('/login', methods=['GET','POST'])
25  def login():
26      form = LoginForm()
27      if form.validate_on_submit():
28          user = db.session.query(User).filter(User.id==form.id.data).one()
29          if user.password == form.password.data:
30              login_user(user)
31              return redirect(url_for('list_users'))
32          else:
33              return render_template('login_failed.html')
34      else:
35          return render_template('login.html', form=form)
36
37  @app.route("/logout")
38  @login_required
39  def logout():
40      logout_user()
41      return redirect(url_for('login'))
```


MSU DENVER

Flask-Login + Google

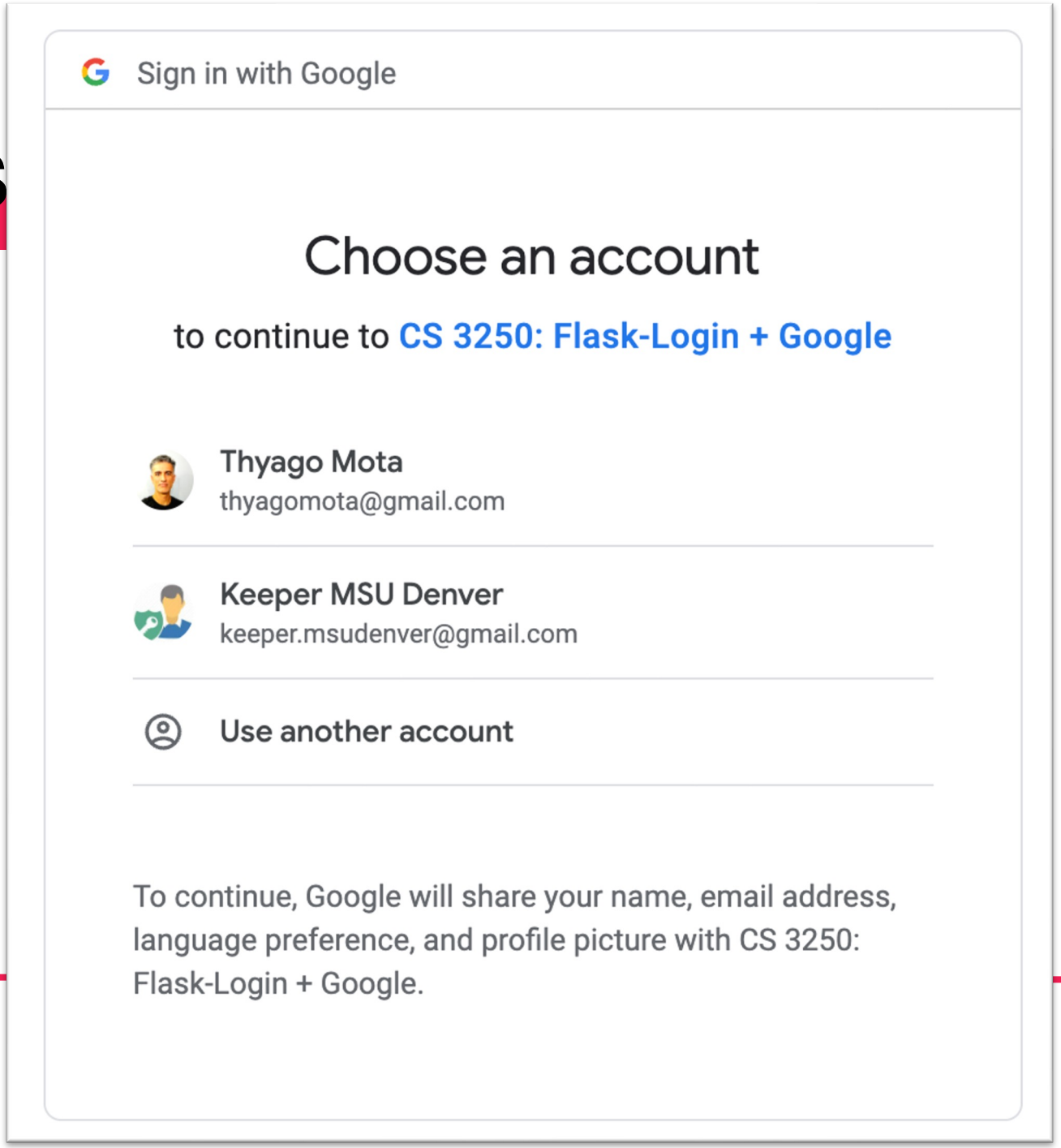
- Google Cloud:
  - Go to https://console.cloud.google.com/ and create a new project
  - Need to enter credit card!

Flask-Login + Google

- Google OAuth 2.0 Client ID:
  - Name: CS 3250: Flask Google Login
  - Origins: https://127.0.0.1:5000
  - Redirects: https://127.0.0.1:5000/login/callback

MSU DENVER

Flas



MSU DENVER

Flask-Login + Google

- Generate a self-signed Certificate:

```
openssl req -x509 -newkey
rsa:4096 -nodes -out cert.pem -
keyout key.pem -days 365
```


MSU DENVER

Flask-Login + Google

```
| run.py
| cert.pem
| key.pem
|___ app
|  |___ __init__.py
|  |___ routes.py
|  |___ models.py
|  |___ forms.py
|  |___ templates
|  |  |___ users.html
|  |  |___ login.html
|  |  |___ login_failed.html
```


MSU DENVER

Flask-Login + Google

run.py

```
1    from app import app
2
3    app.run(ssl_context=('cert.pem', 'key.pem'))
```


MSU DENVER

Flask-Login + Google

```
# OAuth 2 client setup
from oauthlib.oauth2 import WebApplicationClient
GOOGLE_CLIENT_ID = '[redacted]'
oauth_client = WebApplicationClient(GOOGLE_CLIENT_ID)
```


MSU DENVER

Flask-Login + Google

```
GOOGLE_DISCOVERY_URL = 'https://accounts.google.com/.well-known/openid-configuration'
def get_google_provider_cfg():
    return requests.get(GOOGLE_DISCOVERY_URL).json()
```


MSU DENVER

Flask-Login + Google

```
@app.route('/login', methods=['GET','POST'])
def login():
    # Find out what URL to hit for Google login
    google_provider_cfg = get_google_provider_cfg()
    authorization_endpoint = google_provider_cfg["authorization_endpoint"]

    # Use library to construct the request for Google login and provide
    # scopes that let you retrieve user's profile from Google
    request_uri = oauth_client.prepare_request_uri(
        authorization_endpoint,
        redirect_uri=request.base_url + "/callback",
        scope=["openid", "email", "profile"],
    )
    return redirect(request_uri)
```


MSU DENVER

Flask-Login + Google

```
44  @app.route("/login/callback")
45  def callback():
46      # Get authorization code Google sent back to you
47      code = request.args.get("code")
48
49      # Find out what URL to hit to get tokens that allow you to ask for
50      # things on behalf of a user
51      google_provider_cfg = get_google_provider_cfg()
52      token_endpoint = google_provider_cfg["token_endpoint"]
53
54      token_url, headers, body = oauth_client.prepare_token_request(
55          token_endpoint,
56          authorization_response=request.url,
57          redirect_url=request.base_url,
58          code=code
59      )
```


MSU DENVER

Flask-Login + Google

```
63      token_response = requests.post(
64          token_url,
65          headers=headers,
66          data=body,
67          auth=(GOOGLE_CLIENT_ID, GOOGLE_CLIENT_SECRET),
68      )
69
70      # Parse the tokens!
71      oauth_client.parse_request_body_response(json.dumps(token_response.json()))
```


MSU DENVER

Flask-Login + Google

```
73      # Now that you have tokens (yay) let's find and hit the URL
74      # from Google that gives you the user's profile information,
75      # including their Google profile image and email
76      userinfo_endpoint = google_provider_cfg["userinfo_endpoint"]
77      uri, headers, body = oauth_client.add_token(userinfo_endpoint)
78      userinfo_response = requests.get(uri, headers=headers, data=body)
```


MSU DENVER

Flask-Login + Google

```
80      # You want to make sure their email is verified.
81      # The user authenticated with Google, authorized your
82      # app, and now you've verified their email through Google!
83      if userinfo_response.json().get("email_verified"):
84          unique_id = userinfo_response.json()["sub"]
85          users_email = userinfo_response.json()["email"]
86          picture = userinfo_response.json()["picture"]
87          users_name = userinfo_response.json()["given_name"]
88          user = User(
89              id=unique_id, name=users_name, email=users_email, profile_pic=picture
90          )
91          return f'{user.id}, {user.name}, {user.email}, {user.profile_pic}'
92      else:
93          return "User email not available or not verified by Google.", 400
```


MSU DENVER

Flask-Login + Google

- Once you get the user authenticated from Google, you can save the user's info in your web app's local database
- You should also call Flask-Login's login_user to save the user information to the session

MSU DENVER

Flask-Login + Google

- Credits:
  - Create a Flask Application With Google Login – Real Python

MSU DENVER