Flask and Python

- **Question: How can we send information and access it?**
- HDDP Request:
  - Get req → Loading a website, retrieving the website + HTML
  - Post req → Making some change to a database or state of the system, we post the fact that we are doing, so we post all the information that we wanna send.
  - Put req
  - Delete
  - Update
- These requests distinguish what type of methods you are sending.
- We wanna make sure that the login and the sign up accept these post/get requests:
- SO we do: `@auth.route('/login', methods=['GET', 'POST'])`
- Now we wanna get the data now after we login:
  - So we do:

```
def login():
    data = request.form
    print(data)
    return render_template("login.html")
```

Output: ImmutableMultiDict([('email', 'adrinacad.esf@gmail.com'), ('password', '1234')])

- So now we have sent the information.
- **Question: How can we store the information in a database?**
  - We get info in sign up → So we go to the sign-up method
  - If the method is POST:
  - Then we get:
  - password1, password 2, firstname and email.
- Code:

```
def sign_up():
    if request.method == 'POST':
        email = request.form.get('email')
        firstName = request.form.get('firstName')
        password1 = request.form.get('password1')
        password2 = request.form.get('password1')
```

- **Question: How can we check if the information is valid and how to flash erorrs?**
- We write some simple if statements to check if we are doing it right or not

```
if len(email) < 4:
        pass
    elif len(firstName) < 2:
        pass
    elif len(password1) != len(password2):
        pass
    else:
        # The data is right and we can enter it to our database.
        pass
```

- If it was wrong → we send “Message Flashes.”
    - 1. Import the Flash library
    - 2. Then flash the messages.

```
if len(email) < 4:
        flash('Email must be greater than 4 characters.', 
category='error')
    elif len(firstName) < 2:
        flash('First name must be greter than 1 characters.', 
category='error')
    elif len(password1) != len(password2):
        flash('Passwords don\'t match', category='error')
    elif len(password1) < 7:
        flash('Password should be at least 7 characters', 
category='error')
        # The data is right and we can enter it to our 
database.
    else:
        flash('Accounts created!', category='success')
```

    - Each message has: message + category

- **Question: How should we flash the message and the errors on the screen?**
- 1. We first get the messages.
- 2. Use the function “with” → Similar to let x = …. In JS
    - {% with messages = get_flashed_messages(with_categories=true) %}
    - Now in the messages we have: message + category
    - Ex: [("error", "Email must be greater than 4 characters."), ("success", "Account created successfully!")]
    -
- 3. Then we go through the messages if the exists, so we have:
- 4. And we display it on teh web with the html elements:
- **{% if messages %}**
- **{% for category, message in messages %}**
  <div class=“messgae_alert”>
  {{ message }}
  <button>X<\button>
  <\div>
  { % endfor %}
- **{% endif %}**

DATABASE:
Question: How does the database work?

1. Import the right libraries
2.
```
from flask_sqlalchemy import SQLAlchemy
```
3. Then we create the database object:
    a.
    ```
    db = SQLAlchemy()
    ```
4. We give it a name:
    a. DB_NAME = “database.db”
    b. This is the object when we want to add a user, create, delete info and more.

Question: Now we should tell Flask that we are using this database, and where it is located?

- Store the database in the website folder:
-
```
app.config['SQLALCHEMY_DATABASE_URI'] = f'sqlite:///{DB_NAME}'
```

- Now initialize the database by giving it
-
```
db.init_app(app)
```

Question: How to define a database model?

1. We first go to the model.py
2. We import the db object: from . import db
3. Now we need a user class to store different databases:

```
-   class User(db.Model, db.userMixin):
```

4. Now we create the info we want to save from each user, and we make sure that they are unique → Primary key:

```
-       id = db.Column(db.Integer, primary_key=True)
        email = db.Column(db.String(150), unique=True)
```

   - See more example in the file

Question: How can we associate different datas to different users?
Foreign key idea → Read more in the coding file in model.py

Question: What is the next step?

- Ok now we need to crete the database. Steps:
  1. First we import the database in the __init__ file:

```
a.  from .models import User, Note
```

  2. And then crete the crete_database function if we have no database
  3. Then we run it and we see we have the database

Question: How we should we create a user now that we have the database?"
Go to auth.py, in the else section where all the conditions are checked we should have: and import the necessary stuff in the file as well.

```
# Making a user:
            new_user = User(
                email=email,
                firstName=firstName,
                password=generate_password_hash(password1,
method='pbkdf2:sha256')
            )
            db.session.add(new_user)
            db.session.commit()


            flash('Accounts created!', category='success')

            # Now redirect to the home_page
```

```
        return redirect(url_for('views.home'))
```