Mobile Platform - API Connection I

CS2B01 - Desarrollo Basado en Plataformas - Unidad 3

Dr. Jesus Bellido
jbellido@utec.edu.pe

UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

Logros

Unit Outcomes

Al finalizar esta unidad usted estará en la capacidad de:

- Design and implement a mobile application for a given mobile platform [Familiarity]

Unit Outcomes

Al finalizar esta unidad usted estará en la capacidad de:

- Design and implement a mobile application for a given mobile platform [Familiarity]
- Discuss the constraints that mobile platforms put on developers [Familiarity]

Unit Outcomes

Al finalizar esta unidad usted estará en la capacidad de:

- Design and implement a mobile application for a given mobile platform [Familiarity]
- Discuss the constraints that mobile platforms put on developers [Familiarity]
- Discuss the performance vs power tradeoff [Familiarity]

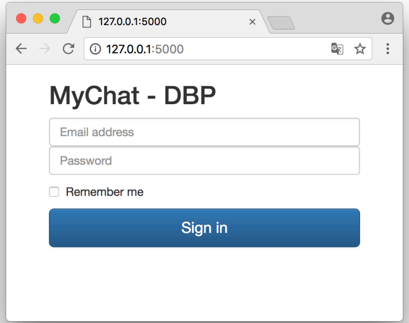
Unit Outcomes

Al finalizar esta unidad usted estará en la capacidad de:

- Design and implement a mobile application for a given mobile platform [Familiarity]
- Discuss the constraints that mobile platforms put on developers [Familiarity]
- Discuss the performance vs power tradeoff [Familiarity]
- Compare and Contrast mobile programming with general purpose programming [Familiarity].

Introducción

Something is missing...


127.0.0.1:5000

MyChat - DBP

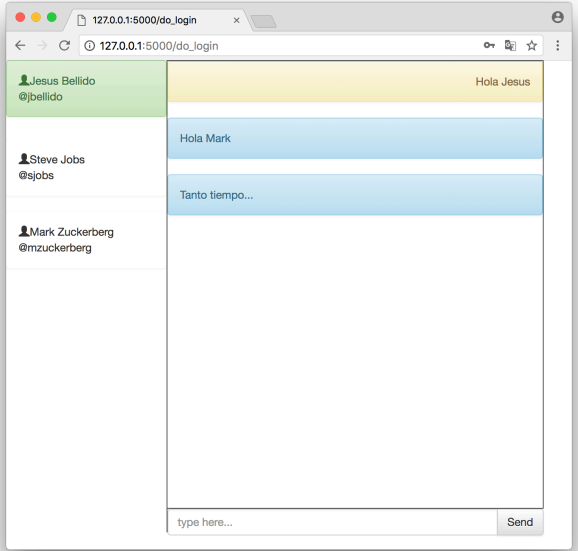
Email address

Password

Remember me

Sign in

Something is missing...



127.0.0.1:5000/do_login

Jesus Bellido
@jbellido

Steve Jobs
@sjobs

Mark Zuckerberg
@mzuckerberg

Hola Jesus

Hola Mark

Tanto tiempo...

Entities

User entity

```python
class User(connector.Manager.Base):
    __tablename__ = 'users'
    id = Column(Integer, Sequence('user_id_seq'),
        primary_key=True)
    name = Column(String(50))
    fullname = Column(String(50))
    password = Column(String(12))
    username = Column(String(12))
```

entities.py

Message entity

```python
class Message(connector.Manager.Base):
    __tablename__ = 'messages'
    id = Column(Integer, Sequence('message_id_seq'), primary_key=True)
    content = Column(String(500))
    sent_on = Column(DateTime(timezone=True))
    user_from_id = Column(Integer, ForeignKey('users.id'))
    user_to_id = Column(Integer, ForeignKey('users.id'))
    user_from = relationship(User, foreign_keys=[user_from_id])
    user_to = relationship(User, foreign_keys=[user_to_id])
```

entities.py

Message Uniform Interface

Message Uniform Interface

```python
@app.route('/messages', methods = ['GET'])
def get_messages():
    session = db.getSession(engine)
    messages = session.query(entities.Message)
    data = []
    for message in messages:
        data.append(message)

    return Response(json.dumps(
      data,
      cls=connector.AlchemyEncoder),
      mimetype='application/json'
    )
```

server.py

Message Uniform Interface

```python
@app.route('/message/<id>', methods = ['GET'])
def get_message(id):
    session = db.getSession(engine)
    messages = session.query(entities.Message).
        filter(entities.Message.id == id)
    for message in messages:
        js = json.dumps(message, cls=connector.
            AlchemyEncoder)
        return  Response(js, status=200, mimetype='
            application/json')

    response = { "status": 404, "message": "Not
        Found"}
    return Response(response, status=404, mimetype=
        'application/json')
```

server.py

Message Uniform Interface

```python
@app.route('/messages', methods = ['DELETE'])
def delete_message():
    id = request.form['key']
    session = db.getSession(engine)
    messages = session.query(entities.User).filter(
        entities.User.id == id)
    for message in messages:
        session.delete(message)
    session.commit()
    return "Deleted Message"
```

server.py

Message Uniform Interface

```python
@app.route('/messages', methods = ['POST'])
def create_message():
    #c =  json.loads(request.form['values'])
    c = request.get_json(silent=True)
    session = db.getSession(engine)
    user_from = session.query(entities.User).filter
        (entities.User.id == c['user_from']['id']).
        first()
    user_to = session.query(entities.User).filter(
        entities.User.id == c['user_to']['id']).
        first()
    message = entities.Message(
         content = c['content'],
         user_from = user_from,
         user_to = user_to,
         sent_on = datetime.datetime.utcnow()
    )
```

More... Uniform Interface

Getting chats

```python
@app.route('/chats', methods = ['GET'])
def get_chats():
  sessiondb = db.getSession(engine)
  user_id = session['logged']
  chats = sessiondb.query(entities.Message.
      user_to_id).filter(entities.Message.
      user_from_id == user_id).distinct()
  data = []
  for message in chats:
    user = sessiondb.query(entities.User).filter(
        entities.User.id == message[0]).first()
    data.append(user)
    ...
```

server.py

Getting chats

```python
@app.route('/chats', methods = ['GET'])
def get_chats():
  ...
  chats = sessiondb.query(entities.Message.
      user_from_id).filter(entities.Message.
      user_to_id == user_id).distinct()
  for message in chats:
    user = sessiondb.query(entities.User).filter(
        entities.User.id == message[0]).first()
    if user not in data:
      data.append(user)

  return Response(json.dumps(data, cls=connector.
      AlchemyEncoder), mimetype='application/json')
```

server.py

Doing login

```python
@app.route('/do_login', methods = ['POST'])
def do_login():
  username = request.form['username']
  password = request.form['password']
  sessiondb = db.getSession(engine)
  user = sessiondb.query(entities.User).filter(
    and_(entities.User.username == username,
        entities.User.password == password )
    ).first()
  data = []
  if user != None:
    session['logged'] = user.id;
    return render_template("chats.html")
  else:
    return render_template("login.html")
```

server.py

Who is logged in?

```
@app.route('/current', methods = ['GET'])
def current():
  sessiondb = db.getSession(engine)
  user = sessiondb.query(entities.User).filter(
      entities.User.id == session['logged']).first
      ()
  js = json.dumps(user, cls=connector.
      AlchemyEncoder)
  return Response(js, status=200, mimetype='
      application/json')
```

server.py

More... Code on Demand

Who is logged in...

```
$.getJSON("/current", function(data){
  current_id = data['id']
  f = '<div class="alert alert-success" ';
  f = f + 'role="alert">';
  f = f+' <span class="glyphicon glyphicon-user"
      aria-hidden="true"></span>';
  f = f+ data['name']+" "+data['fullname'] ;
  f = f+'<div>@'+data['username']+'</div>';
  f = f+' </div>';

  $( "<div/>", {
    html: f
    }).appendTo( ".top" );
  });
```

chat.html

My Chats?

```
$.getJSON("/chats", function(data){
  var items = []
  i = 0
  $.each(data, function(){
    f = '<div class="alert " ';
    f = f +  "onclick = verChat("+data[i]['id']+")"
    f = f + 'role="alert">';
    f = f+' <span class="glyphicon glyphicon-user" aria-hidden="true"></span>';
    f = f+ data[i]['name']  +" "+ data[i]['fullname'];
    f = f+ "<div>@"+data[i]['username']+"</div>";
    f = f+' </div>';
    i = i+1;
    ...
 });
```

My Chats?

```
$.getJSON("/chats", function(data){
  ...
  i = i+1;
  $( "<div/>", {
    html: f
    }).appendTo( ".left" );
 });
});
```

chat.html

My Messages?

```
function verChat(id1){
  key = current_id+","+id1;
  $( ".right" ).empty();
  $.getJSON("/chats/"+key, function(data){
    var items = []
    i = 0
    $.each(data, function(){
      if(data[i]['user_from_id'] == current_id){
        f = '<div class="alert alert-info" role="alert">';
      } else {
        f = '<div class="alert alert-warning" style="text-align: right;" role="alert">';
      }
      f = f+ data[i]['content'] ;
      f = f+'</div>';
      ...
    })
```

My Messages?

```
function verChat(id1){
  key = current_id+","+id1;
  $( ".right" ).empty();
  $.getJSON("/chats/"+key, function(data){
    var items = []
    i = 0
    $.each(data, function(){
      ...
      $( "<div/>", {
        "class":"message",
         html: f
      }).appendTo( ".right" );
      i = i+1;
    });
 });
}
```

chat.html

Conclusion

Summary

Get the source of this demo presentation from

`github.com/CSUTEC-CS2B01-B`

Questions?

Abstract

References I