Introduction to Processes of Object-Oriented Software

COP 4331
University of Central Florida
Michael McAlpin

Instructor Contact

- Email: Michael.McAlpin@ucf.edu
- Office Hours:
  - Tuesday & Thursday: 13:00 – 15:00 HEC-318
  - Monday and Wednesday, by appointment via Zoom

Course Objectives

- This is a software engineering course

- Understand software engineering
- Create and demonstrate a small, full-stack project
- Create and demonstrate a large, full-stack project with a mobile component

This Course is Unlike Those You've Already Had

- In courses such as Intro to C, Computer Science 1, Computer Science 2, and Programming Languages most students start their assignments at 1PM on Sunday and turn them in by 11:59PM.
- That is not the way this class works.
- To be successful you need to spend time throughout the week working on your classwork and team responsibilities.

Teams

- You have been assigned to a team of five members.
- Normally you would be able to select your own team,
  - First project: Assigned to teams to align with Friday Labs.
  - Large project: You can sign up with your own team if you choose.
- In most cases, everyone on the team gets the same grade for projects. But there are exceptions for team members who do not contribute their fair share.
- You are expected to have weekly meetings during labs, if possible,
  - Maintain team communications using tools such as discord or slack.
- You should take personal responsibility and do your part. In senior design we have a saying “Do what you say you will do” or DWYSYWD.
- Someone on the team should function as the project manager.

Team Functioning

- Teams should split the development responsibilities according to who has the best skills in a give area.
- For instance, most groups separate development concerns for the small project to database, API, front end, and project management.
- Please note that we have a large class, and therefore large groups. There should probably be two people of the front end, two on the API, one on the database, and one serving as project manager.
- Whenever possible, there should be some overlap between functions.

Learning

- What if nobody on your team knows about databases, or APIs, or front ends?
- You will learn the basics in class, but these are just “getting started” lectures.
- You will need to spend time learning on your own with YouTube videos and online courses.
- You can also visit me during office hours (T/R 13:00 – 15:00)
- I am also willing to schedule extra zoom sessions.
- In this class, you will learn a lot more than I have time to teach you.

The Small Project

- The small project is a contact manager.
- When a user first accesses the website, they can either login or sign up (register).
- Once they are logged in with a valid account, they can perform standard CRUD operations on their list of contacts.
  - C – Create
  - R – Read
  - U – Update
  - D – Delete
- You will also need to add a search function for contact records.

Remote Server

- Your web app **must** reside on a remote server.
  - (Not a computer in your basement, garage, friend's house, attic, etc...)
- Good choices are GoDaddy, Heroku, Digital Ocean, AWS, and Azure.

Contact Information

- The **minimum** contact information your app needs for each contact record is:
  - Name (prefer first name/last name)
  - Email
  - Phone
  - Date record created.

Small Project Requirements

- Professional PowerPoint slides.
- Project works with login/register and CRUD operations.
- There must be a search function where users can search for specific contacts.
- All members must participate in the presentation.
  - Not just saying their name
    - but making a substantial contribution.
- You must use a LAMP stack. (No Python)
- You must communicate from the front end to the backend through an API.
- Your PowerPoint slides must include a Use Case diagram and a Gantt chart.
- You must demonstrate at least one API endpoint with SwaggerHub.
- Your PowerPoint slides must include an entity relationship diagram (ERD).
- Postman API tools

Your Responsibility

- Do not ghost your team.
- You need to participate in the weekly meetings.
- You need to do your part – do what you say you will do (DWYSWD).
- It is not fair to your team if you are a social loafer.
- Your grade will also suffer – many people think they can hide in a group and take advantage of their group's efforts.
- If you have a full-time job, I recommend dropping this class. This is a 4000-level class that prepares you for senior design. It takes time and dedication.

GitHub

- All of your team's code must be in a GitHub repository.
- It needs to be a public repository so that the TA and I can access it and check your progress.
- You will need to add the GitHub link to the project spreadsheet.
- GitHub will allow you to safely share code with your team.
- It also provides a way for the instructor to spot check your code.

User Experience

- The small project is a web app.
- You should use best practices for the user experience.
- The only alert/dialog that should appear is a delete confirmation.
- Test your app on different devices including full-screen desktops and phones.

API Documentation

- You will need to use Postman (postman.com) to document and test your API.
- You will need to show at least one endpoint during your presentation.
- The API developers will need to keep it updated so that the team can see how to use the API and any changes that may have been made.

PowerPoint Presentation Submission

- All members of the team must submit the PowerPoint slides to the WebCourses.
- Do not zip the file.
- Make sure you submit the file before you group does its presentation.

Diagrams

- You are required to include the following in your PowerPoint presentation:
  - Gantt chart
  - Use Case diagram
  - Entity Relationship Diagram

Loading All Recordsets???

- Do not load all record sets into memory.
- If you had 10,000 contacts it would take a long time to retrieve them.
- If a user types in **Jo** the search algorithm should match everything with Jo in it (*case insensitive*) i.e: John, Jones, Jobs
- You should be able to search on first and last names at the very least.

Again…

- Do not ghost your team.
- You are taking a 4000-level computer science course at a major university.
- This is not psychology 101.
- You need to perform up to the level of the course.
- Later courses, especially senior design rely on what you learn in this class.

Suggested Roles

- One person should act as a project manager. (PM)
  - They can also develop in order to provide some overlap.
  - This person is responsible for helping to keep team members on track.
- One person should maintain the database. (DBA)
  - This person needs to create an entity relationship diagram for the members of the team.
  - The ERD will also become part of the PowerPoint presentation.
- Two people should develop the API endpoints. (API Dev)
  - Each endpoint must be represented and testable on Swagger Hub so that the front end developers can easily check how to use each API endpoint.
- Two people should develop the front end. (Front End Dev)
  - They should consult with the Swagger Hug documentation.
  - It is suggested that bootstrap and jquery is used to create a nicer user experience.
  - Don’t use alert boxes except during testing – the exception is for delete confirmation boxes.
- Anyone can get hit by a bus (MBA)
  - It is recommended that overlap development responsibilities in the unlikely even that someone legitimately cannot perform their duties.

Questions

APIs to the Rescue

- In order to access data from a location not controlled by the vendor, you will need to create and use a REST API. (REST stands for Representational State Transfer)
- A REST API allows communications through port 80 or 443, both of which most web servers support, and firewalls allow to pass through.
- ***NOTE: 80 and 443 are not magic numbers but these are the ports that web servers use. You could in some circumstances use other ports.***

Application Programming Interface -- An API is a set of functions and procedures allowing the creation of applications that access the features or data of an operating system, application, remote database, or other service.

The REST API In Action

From your desktop computer you want to know how many red widgets are in your company's inventory.

Continued

Your system packages up a request.
More about how it's formatted later.

Continued


Package travels to
server via HTTP protocol
on port 80 or 443

Query Time

Server Performs Query in
Response to Your Request

Response Time

Now the Client (Requester) Has the Data

- Now the client has the data and can do useful things.
- The code running on the server that performs tasks for the requester is normally referred to as an endpoint.
- Sometimes there is business logic performed in the API endpoint. These endpoints were referred to as middle tiers in the early 2000s.

Packaging Syntax

- There are two major syntaxes used in created request and response packages.
- By far the most common is JSON (JavaScript Object Notation).
- Far in the rear-view mirror is XML.
- It is possible to create your own format, but this is not advised.

How is JSON Formatted

JSON opens and closes with curly braces.

```
{
}
```

Data inside are key/value pairs.

```
{
  "Name" : "Rick",
 "City" : "Orlando",
 "DoubleValue"  : 45.7,
 "IntValue" : 12
}
```

JSON Arrays

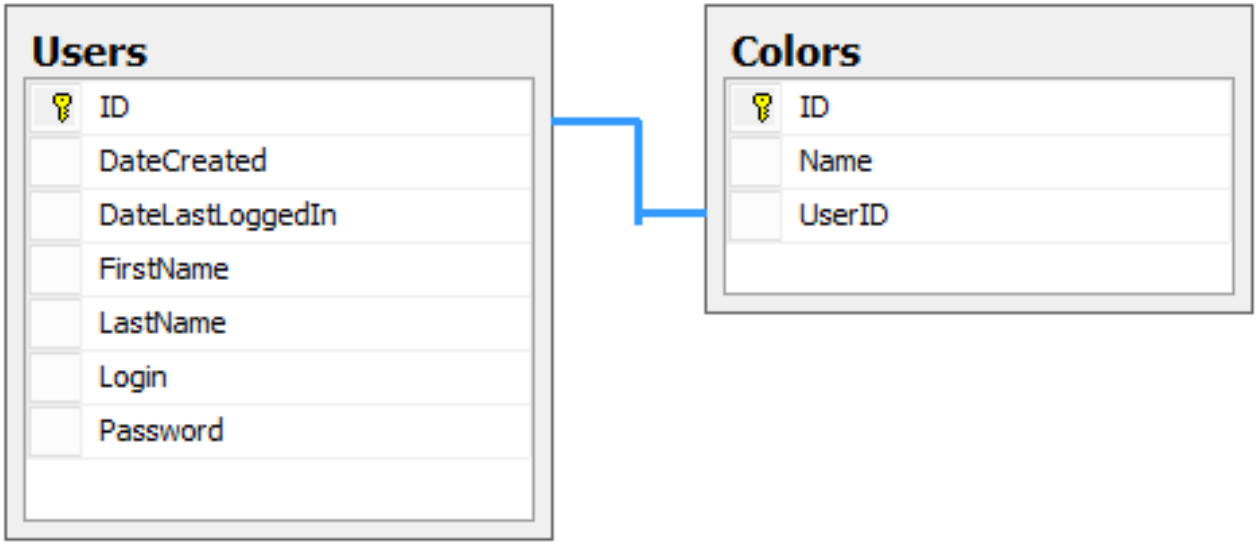
- JSON accommodates arrays.

```
{
  "MyColors" : ["red","green","blue"]
}
```

And arrays of objects.

```
{
  "MyContact" : [
  {"Address" : "1234 Main Street", "City" : "Orlando"},
  {"Address" : "456 Elm Road“, "City“ : "Cincinnati"},
  {"Address" : "10321 SW  This St.", "City“ : "Miami"},
  ]
}
```

Database Relationships



Creating JSON From Front End

- Suppose you have an HTML form with two fields: login and password (the form names are **loginName** and **loginPassword**)

- Get the UI values as follows:

```
var login = document.getElementById("loginName").value;
var password = document.getElementById("loginPassword").value;
```

Turn the Into JSON

```
var jsonPayload = '{"login" : "' + login + '", "password" : "' + hash + '"}';
```

Alternatively, you can use stringify:

```
function doIt(nm,pw)
{
        var d = {login:nm, password:pw};
        var json = JSON.stringify(d);
        // Now do something with the JSON
}
```

Send the JSON to the Endpoint

- Once you have the JSON you can send it to the endpoint

```
var xhr = new XMLHttpRequest();
xhr.open("POST", url, false);
xhr.setRequestHeader("Content-type", "application/json; charset=UTF-8");
xhr.send(jsonPayload);
```

Get the Results

- Convert the returned object into JSON and get the values.

```
var jsonObject = JSON.parse( xhr.responseText );
userId = jsonObject.id;
firstName = jsonObject.firstName;
lastName = jsonObject.lastName;
```

API Endpoints