CSI3140 Project Winter 2022

Over the semester you will be building a web application that will incorporate one or more of the following:

- HTML (or similar like Erb, EEX, Elm)
- CSS / SASS
- JavaScript
- Server-side scripts such as PHP, Elixir, Go, or Node
- Internal APIs (RESTful using JSON HTTP, GraphQL, or Web Sockets)
- External APIs (e.g. google maps, stripe payments, etc)

You will be using git to track progress over the semester, and your ability to consistency make progress towards will be factored into your overall mark.

The term project can be done in a group of 2. If you choose to go solo, the scope of the project will not change. Groups of 3 are possible but discouraged. It is the team's responsibility to properly manage their time.

You must submit a project to satisfy the requirements of the course.

Track all Artifacts

All code, all images, all content should be in your GitHub repository.

If some artifacts are derived (i.e. compiled) then include instructions on how to build those artifacts AND include a copy of them (as a backup should the build process not work).

Deliverables

The project will be worth 30% of your final grade, divided up as follows

Deliverable 1 (3 marks) Project Definition

Mark	Description
2.0	GitHub repository setup
2.0	List team members including names, and student numbers
3.0	Outline the application you will be building
2.0	Proper access granted to TA and professor
1.0	Git usage (commit messages, all students involved)
/ 10	

You will create your team, pick a project to develop and get familiar with Git, GitHub and Pull-Requests.

For this delivery you will be responsible for

- Creating your project team of 2 ± 1 persons
- Creating your GitHub Classroom repository
- README.md with you team members (names, student numbers) and description of your application

Deliverable 2 (7 marks) Project Mock-Ups

Mark	Description
3.0	Mock-Ups in HTML / CSS
1.0	Colour Palette
1.0	Fonts and Type Scale
1.0	Icons (and other images)
1.0	Buttons and Form Elements
1.0	UI Components (e.g. popups)
2.0	Updated README.md to document UI Design System
1.0	Git usage (commit messages, all students involved)
/ 10	

You will create HTML / CSS mockups of your application. You will also build (and document) a UI Design System to highlight the set of standards for your colour palette, fonts, icons, buttons, UI components and form elements.

Deliverable 3 (10 marks) Technology Landscape

Mark	Description
2.0	Server Technology integrated (e.g. PHP, Elixir, Go) including library and frameworks
2.0	Database Technology integrated (e.g. MySQL, Postgres, Redis, etc)
1.0	Automated test framework in place
2.0	Deployment / Upgrade Scripts working
1.0	Refined HTML/CSS + UI Design System
1.0	Front-end (mock) interactivity using JavaScript
1.0	README.md updated with installation / deployment instructions
/ 10	

You will pick a technology stack and implement a "hello world" application showing how all pieces fit together.

The simplest tech stack for your application is probably PHP (which has a built-in webserver) and PostgreSQL. You are free to use other technologies like Java, Ruby, Python, Node, Elixir but you will be responsible for clearly documenting how to install your applications on a Linux environment (including Mac OSX).

This could be a great opportunity to learn new technologies but do not expect much support as the professor and TA might not be familiar with your chosen tech stack.

Deliverable 4 (10 marks) Features

Mark	Description
3.0	Implementation of features Server / Client / HTML / CSS
2.0	Software Documentation (installing, testing and developing the
1.0	Adherance to UI Design System
1.0	Seeding application with sample data
1.0	Screenshots of available features
1.0	Application v1.0 (quality versus quantity)
1.0	Git usage (commit messages, all students involved)
/ 10	

With the foundation of your project in place, now it is time to bring that project to life.

This deliverable includes

- Building your web application
- Using Pull-Requests and Code Reviews by all team members
- Maintaining an updated README that promotes your web application
- Refinements to your deployment process
- Deliver V1.0

Other Comments

Ideas

Here are a few ideas to consider

- Online marketplaces like Kijiji, or eBay
- Personal budget software like mint.com
- Stock portfolio manager
- Gym workout software
- Software tools like StatePage.io, ou IsItUp.org
- Online editor like medium.com
- Please no TODO apps
- Architecture, Landscape design applications
- Online IDE for Umple
- Online code editor like repl.it or letscode.ca

Please make it unique and not just a clone of an existing service.

Git

All work should be orchestrated through Git.

We will offer two labs on Git and GitHub and we strongly recommend doing additional practice / online training to ensure that you have a good understanding of how to use git properly.

Commit Messages

Garbage-In, Garbage-Out. You will be evaluated on the quality of your git commit messages and the modularity of what you commit.

DO NOT JUST `git add -A; git commit -m "stuff"`

Here are a few talks, articles, courses about git

- Git for Humans (talk) and slide deck
- Git for Humans (book)
- Fix Common Git Mistakes
- Learn git concepts, not commands
- Git Basics
- Git Rebase
- Git Branching

- git rebase -i HEAD~25 and slide deck
- git log – the Good Parts
- Telling stories with your Git history

Feature Branches

In this project, you will be expected to work in small chunks of work in separate feature branches and you should use pull-requests and code reviews before merging back into the code.

For example, if you were adding a project description, you might create a feature branch as follows

```
git checkout -b f/project-description
git push -u origin HEAD
```

You can then commit and push to that branch.

If your teammates want to contribute to your branch as well, then they would

```
git fetch
git checkout f/project-description
```

COMMUNICATE WITH EACH OTHER about who is doing what. Git cannot stop you editing the same files, in different ways and causing conflicts.

Git Terminal Updates

You might want a script similar to the following to easily tell you which branch you are on

This can within `~/.bash_profile` (or `~/.bash_aliases`) and is known to work on a Mac OSX

```
function parse_git_branch {
  ref=$(git symbolic-ref HEAD 2> /dev/null) || return
  echo "("${ref#refs/heads/}")"
}

RED="\[\033[0;31m\]"
YELLOW="\[\033[0;33m\]"
GREEN="\[\033[0;32m\]"

PS1="$RED\$(date +%H:%M) \w$YELLOW \$(parse_git_branch)$GREEN\$ "
```

Your terminal then might look similar to