Project for UCLA Computer Science 97

[97 home]

For the main project, you will creating a full-stack web app.

Features

Although the app is your choice, it should:

- Display dynamic data to the user. That is, some part of the webpage changes based on data received from the server.
- Upload data from the client to the back-end, which persists (saves) the data to the server's file system. This data can be as simple as strings or numbers that you save to a text file, or files that the user uploads such as images.
- Meaningfully search through server-side data.
    - Examples of meaningful search include searching for certain file names, data content, upload dates, etc.
    - These files could be ones that the user has uploaded, or could be ones that you have placed in the server directory yourself. For example, if you were making an informational app for prospective dog owners you might have a single file containing information about various dog breeds.
    - The front-end must contain let the user enter a query conveniently.

In addition to the above functionality requirements, your web app should contain at least 3 more distinct features. For example, if you were implementing Instagram for the first time, the features might be:

1. Like and comment on photos
2. Visit user profiles that show name, bio, and the user's uploaded photos
3. Follow other users.

If you were implementing the MyUCLA Class Planner, the features might be:

1. Add/remove classes from your current plan.
2. Enroll in classes.
3. List additional information about classes such as days, time, location, instructor, and units.

App ideas include: fitness, social media, cooking, ride-sharing, messaging, music, banking, dating, productivity, COVID-19 tracking with privacy, etc. Don't think about it too hard; you are being evaluated on the quality of your implementation, not the app idea itself! It's totally okay to choose an app idea that already exists.

Though we do not expect your apps to be completely polished and visually aesthetic, we do expect it to be relatively organized and easy to use.

Technology

Node.js and React.

If you would like to try a different technology stack (i.e. build an iOS/Android app, use a different back-end framework such as Python/Django, etc.), you may provided you receive permission from an instructor by the end of Week 6. Your project is valid as long as it meets the requirements outlined above. Please note that choosing a different technology stack may limit the amount of help that instructors can provide you.

You must use Git and GitHub (or similar Git-based repository) to track the progress of your work. The project can receive a full grade only if the submission includes Git history no shorter than five commits from each group member. Commit history may be checked to ensure participation. "Participation" does not mean commits or lines of code, but rather the quality of work.

As is usual in this class, your work should be your own. Of course by using Node.js etc. you will be building on the work of others, and you will be working in a group project, but each of your own commits should be your own work. You should use a private GitHub repository (not visible to outsiders) to help ensure this.

Since you will submit your repository, it should also contain a README file (plaintext or Markdown) describing in detail how someone who cloned it can run your app. This should, of course, include any shell commands needed for setup.

Finally, each student must also submit a personal three-page final report that includes:

- Your app's purpose
- A brief overview of your app's architecture and the technologies you used
- A description of the features supported by your app

- A description of your individual contribution to the project
- Any notable difficulties or challenges you and/or the team faced
- Any improvements or additional features you'd make if you had more time. Improvements can include things that you would do differently if you had the opportunity to build this project again.

See the USENIX templates in Resources for written reports and oral presentations for good report formats.

Grading

Project grading will consist of the following components. All percentages below are percentages of the total project grade.

Group project components (one per group):

- 2% – 1- to 2-page project proposals.
- 2% – 1- to 2-page initial schedule and project plan. This should specify at least one (preferably two) intermediate milestones for the project. The point of the milestones is to have running code early, that implements some project features and can be demonstrated. For an example, see Lucy Deckard's "Developing timelines and milestone charts for your proposal" although we don't expect anything that fancy. Plus, we don't expect the plan to be perfect! It's just a plan, and will no doubt be adjusted as the project evolves.
- 8% – App can display dynamic data to the user.
- 8% – App can upload data from the client to the back-end.
- 8% – User can meaningfully search through server data.
- 24% – Three more distinct features.
- 5% – Meaningful understanding of Git is exemplified through version control.
- 6% – Detailed README file that accurately and completely describes how to run the app locally.
- 6% – Project is generally visually pleasing and easy to navigate.
- 16% – Project presentations and demos (see Resources for advice about presentations).

Final report (one per student):

- 3% – Project description and overview is an accurate account of the project submitted.
- 5% – Report details individual contribution in a meaningful way, and can be verified by git history.
- 4% – Thoughtful reflections of difficulties faced, along with descriptions of how you overcame them.

- 3% – Discussion of improvements/things you would have done differently and additional features you would have liked to implement.

Submission

You will present and demo your project during the course.

One team member must submit a compressed tarball of your GitHub repository to CCLE. This should contain all your group's work, including any documentation, presentations, etc. The repository should not include generated files, only source code; the README file should explain how to generate any files (e.g., node modules) buildable from the source (adding their names to your `.gitignore` file can help you avoid submitting them by mistake). You should be able to obtain such a tarball directly from GitHub. Name the file ***`UID`*`-project.tar.gz`**, where *UID* is your student ID.

In addition, each group member must submit a final report to CCLE individually. Ensure that your report is in PDF format and is named ***`UID`*`-report.pdf`**.