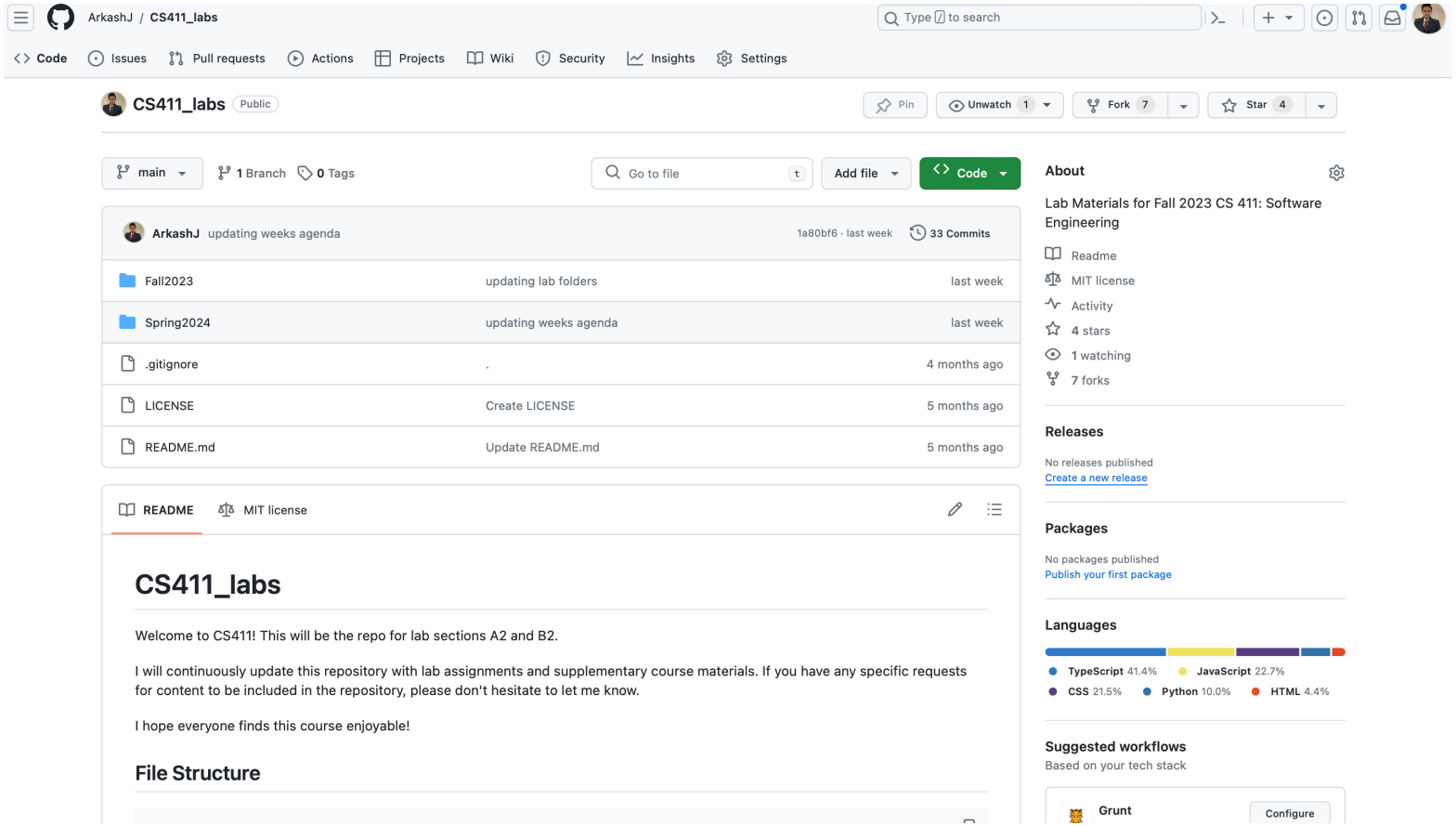
>$ find / -name “teams.md”

Lab 02 – CS 411 @ Boston University

A Quick Recap…

- **Installing Git, Setting up SSH Key, Understanding the Workflow**
  - Git Bash -> Windows
  - Terminal -> macOS, Linux
  - **DO NOT SHARE YOUR PRIVATE KEY**
  - Complete work on <username> branch
    - New branch per feature! *(e.g. `git checkout –b feature/<name>`)*
- **Explore the Lab Repository!**
  - *https://github.com/ArkashJ/CS411_labs/tree/main/Spring2024*
  - Contains any resources shown and/or used during lab
  - If you have suggestions let me know!


ArkashJ / CS411_labs
Type / to search
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
CS411_labs
Public
Pin
Unwatch 1
Fork 7
Star 4
main
1 Branch
0 Tags
Go to file
t
Add file
Code
ArkashJ updating weeks agenda
1a80bf6 · last week
33 Commits
Fall2023 updating lab folders last week
Spring2024 updating weeks agenda last week
.gitignore . 4 months ago
LICENSE Create LICENSE 5 months ago
README.md Update README.md 5 months ago
README
MIT license
CS411_labs
Welcome to CS411! This will be the repo for lab sections A2 and B2.
I will continuously update this repository with lab assignments and supplementary course materials. If you have any specific requests for content to be included in the repository, please don't hesitate to let me know.
I hope everyone finds this course enjoyable!
File Structure
About
Lab Materials for Fall 2023 CS 411: Software Engineering
Readme
MIT license
Activity
4 stars
1 watching
7 forks
Releases
No releases published
Create a new release
Packages
No packages published
Publish your first package
Languages
TypeScript 41.4%
JavaScript 22.7%
CSS 21.5%
Python 10.0%
HTML 4.4%
Suggested workflows
Based on your tech stack
Grunt
Configure

Ladies and Gentlemen…
The Moment You've All Been Waiting For…

ASSEMBLE YOUR TEAMS!

Today's Tasks!

- First, please fill out this form so we know what your teams are!
  - https://docs.google.com/document/d/1I4q6jr8ZGGHIYZEd6wrJv17_xVewuF5dtWIzxwkPwzg/edit?usp=sharing
- Next, I'm going to go over the expectation this semester and answer any questions.
  - I'll also give you time to get familiar with your teammates and establish some methods of communication.
  - Find out what your team can and can't do. In other words, play to each others' strengths!
- Lastly, let's get some ideas down! I'll go over what was done in prior semesters, but it would be a good idea to start thinking now!

User Stories: Bridging Gaps Between Users and Developers

The Role of User Stories in Project Development

- **Definition of User Stories**: Short, simple descriptions of a feature told from the perspective of the user who desires the new capability.
- **Components of a User Story**: Typically includes the type of user, what they want, and why, to ensure the development team understands the user's needs.
- **Benefits of User Stories**: Helps prioritize features based on user needs, fosters better communication and collaboration within the team, and guides the development process.
- **User Stories in Agile**: A key element in Agile methodologies, facilitating flexible, user-centered development and iterative feedback loops.

Photo by Jonathan Borba on Unsplash

Case 1: Build a Photo Editing App

- What features should it have?
  - Undo/Redo Mistakes
  - Crop, brightness – filtering and modification
  - Template feature – show examples of what people can do with the app
  - Import, saving, download
- What actions can the user do?
  - Login to the app
  - Modify pictures
  - Save pictures
- What data do I need?
  - Login – username, password
  - Pictures

CASE 2: Student Link

- Whats wrong here?
  - Overcomplicated, overengineered
  - Hard to navigate
  - Not familiar
- Old Student Link
  - Easy to use

CASE 3: DoorDash

- Talk about the features:
  - Refunds
  - Location
  - Deals
  - Search Bar
  - Easy Navigation

Example

The following user story describes uploading photos to a photo editing application.

User story: As a user, I want to upload and organize my photos		
Scenario: The user connects a camera to their computer, the presence of new photos is automatically detected and they are asked whether they want to upload some or all of those images. After uploading the user can preview each image, add keywords, and decide which ones to keep. When done a new event can be created to organize the new photos to make them easier to find later.		
Actions: Upload photos (all photos or selected photos) Preview photos Delete one or more photos Add keywords to annotate my photos Organize photos into events	**Data:** Photos are tagged with date and time, where they were taken, and any keywords that the user entered. Events are identified by name and an optional description.	**Features or constraints:** New pictures are detected when a camera is connected User will get feedback about how long it will take to upload photos If an internet connection is not available. Photos can be uploaded to a local album on my computer and will be automatically copied to the cloud later.

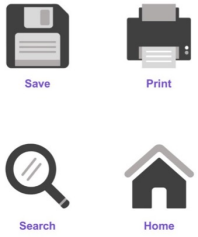
Red Hat | BOSTON UNIVERSITY

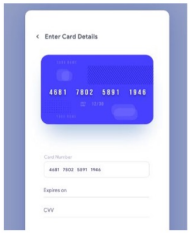
2. Match between system and the real world

The design should speak the users' language. Use words, phrases, and concepts familiar to the user, rather than internal jargon. Follow real-world conventions, making information appear in a natural and logical order.

The way you should design depends very much on your specific users. Terms, concepts, icons, and images that seem perfectly clear to you and your colleagues may be unfamiliar or confusing to your users.

When a design's controls follow real-world conventions and correspond to desired outcomes (called natural mapping), it's easier for users to learn and remember how the interface works. This helps to build an experience that feels intuitive.





Red Hat | BOSTON UNIVERSITY

Make it Familiar and Easy!

Which one would you rather have?

Key Components of a Full Stack Application

- **Decoupled Architecture**: Essential for modern web applications, involving separate front-end and back-end layers that interact through APIs.
- **Front-End Development**: Involves creating the user interface and user experience with technologies like HTML, CSS, and JavaScript frameworks.
- **Back-End Development**: Focuses on server, application, and database management, ensuring data processing and API integration.
- **Integrating External APIs**: Enhances functionality by incorporating services like data.gov for free, public data access.
- **Utilizing a Database**: Stores application data and user information, crucial for features like OAuth user

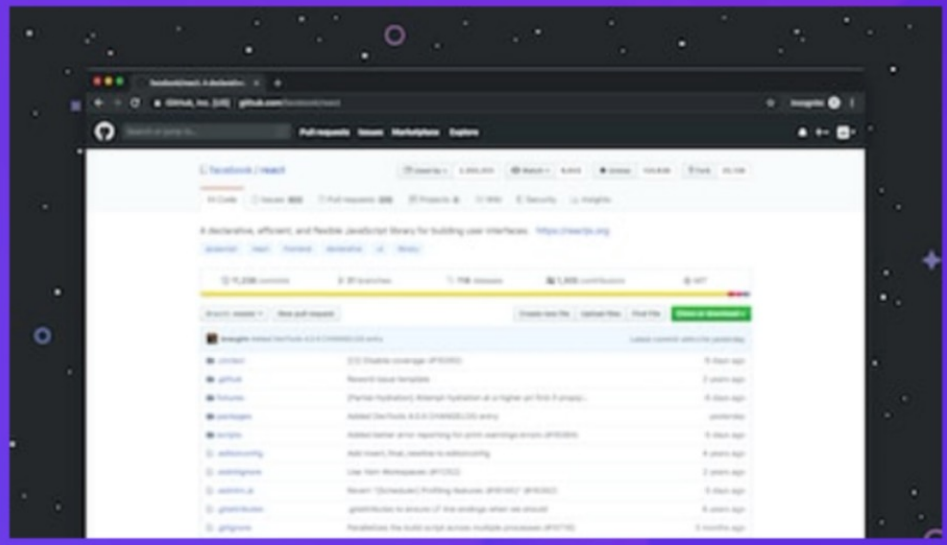
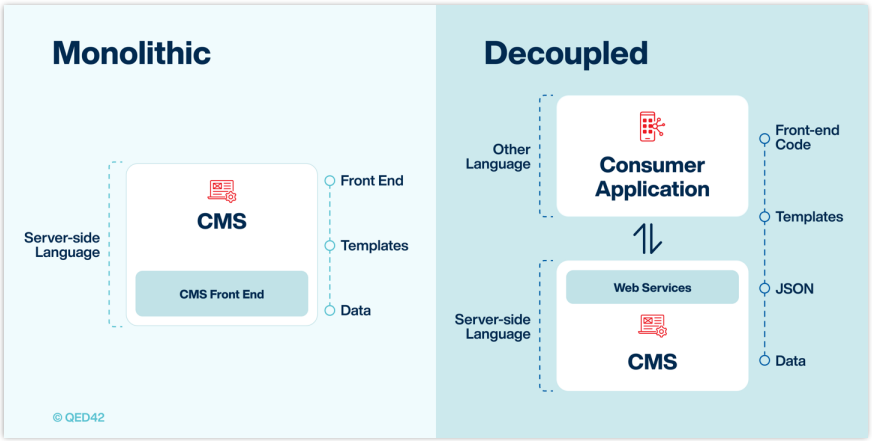


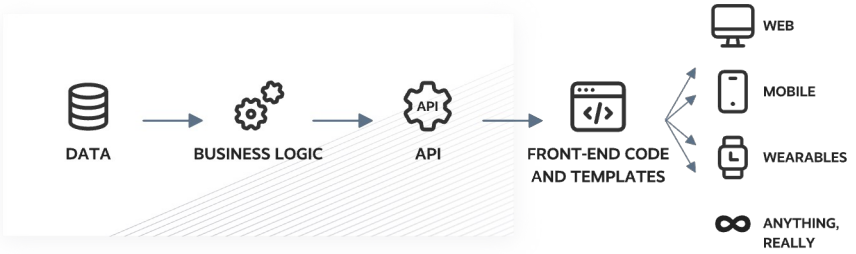
Photo by Luke Chesser on Unsplash

Choosing a stack

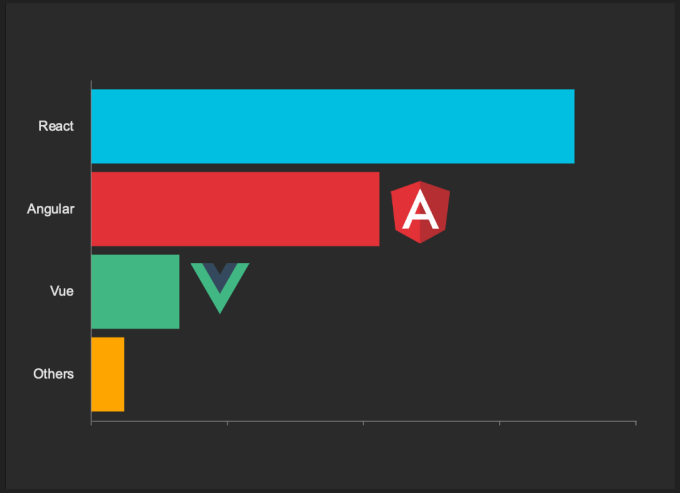
Project Requirements

- It MUST have a decoupled architecture!
  - I.e., It must have a FRONT END and a BACK END
- Make calls to TWO external API's
  - These can literally be ANYTHING you want
  - I personally like https://data.gov/ because it's simple and free!
    - *(Ok well not exactly your tax dollars pay for it…)*
- Utilize a DATABASE, again dealers' choice here
- Have OAuth, meaning allow users to signup and login
  - You actually need a database for this to store credentials





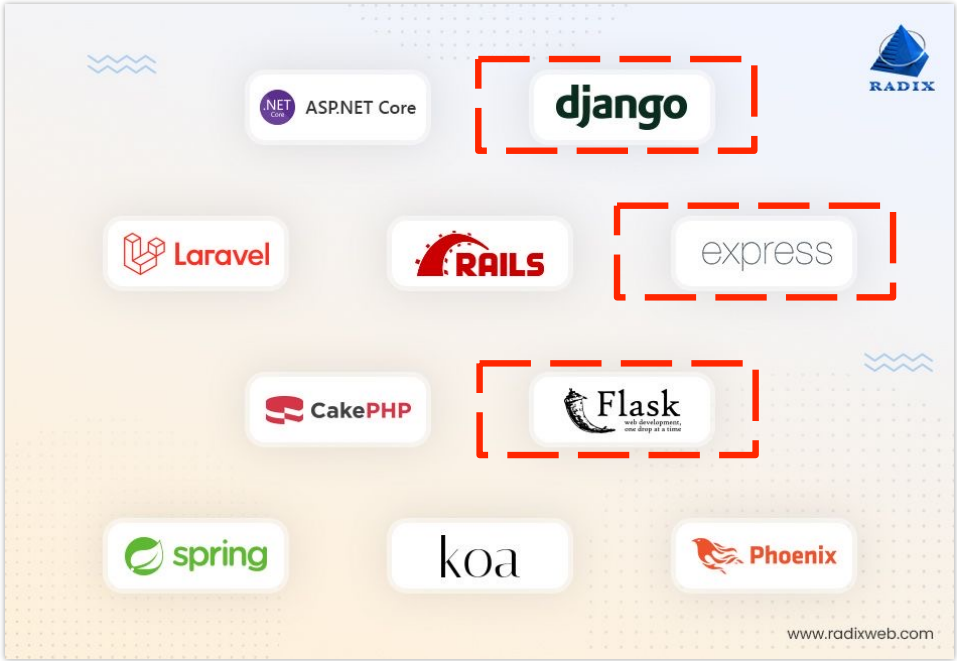
Why is hiding data useful?



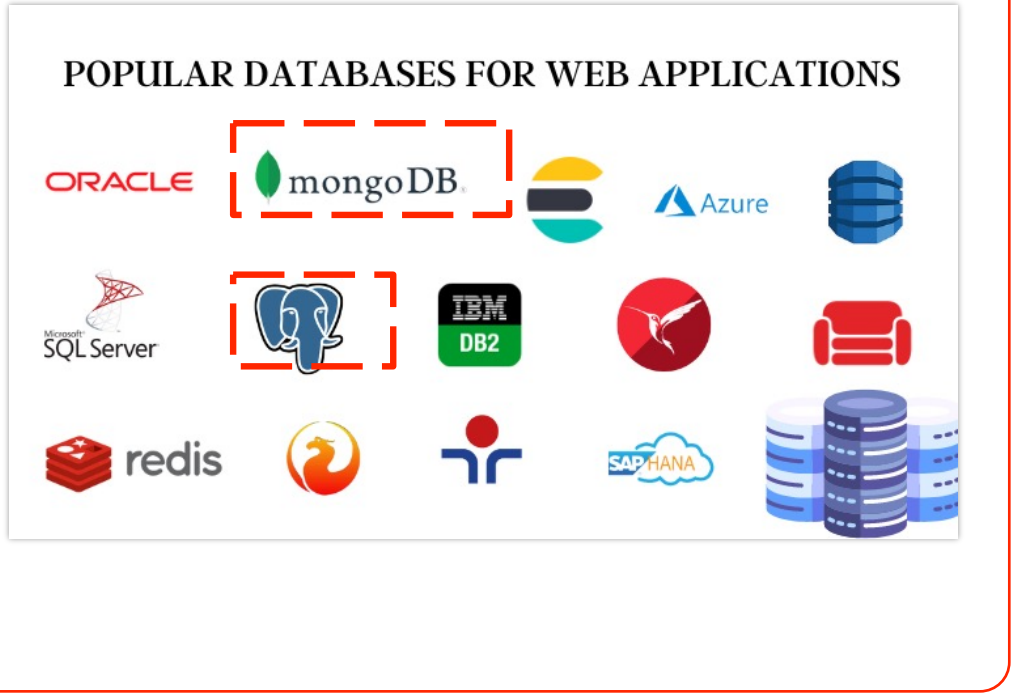
Frontend

- This is what the user sees and interacts with

Backend


RADIX
ASP.NET Core
django
Laravel
RAILS
express
CakePHP
Flask
spring
koa
Phoenix
www.radixweb.com

Database


POPULAR DATABASES FOR WEB APPLICATIONS
ORACLE
mongoDB
Azure
Microsoft SQL Server
IBM DB2
redis
SAP HANA


Get Those Comms Up Soldier!
Signal
slack
Gmail

RESOURCES! *Oh… Uh… And me of course!*

git add lab02.txt
git commit –m “complete”
git push origin master

Lab 02 – CS 411 @
Boston University

(dcmag@bu.edu)