Cheklist to takeoff

Building a node and express app: tying everything together to make a working app.

Create an ERD design (use one to many relationships)* Create user stories on Trello

Must be deployed online to get mongo working on heroku.

- **Run your user stories, wireframes, database design, and an app deployed on Heroku by your instructors**

Necessary Deliverables

- A **working full-stack application, built by you**, hosted on Heroku
- A **link to your hosted working app** in the URL section of your Github repo
- A **git repository hosted on Github**, with a link to your hosted project, and frequent commits dating back to the **very beginning** of the project. Commit early, commit often. Make your commit messages descriptive.
    - Your git repo should have **at least** 30 commits
- **A `readme.md` file** with explanations of the technologies used, the approach was taken, unsolved problems, and notes to yourself so you can come back to your project later in the course and be able to pick up your train of thought, etc.
- **Wireframes of your app**, hosted somewhere & linked in your readme
- A link in your `readme.md` to the **PUBLICLY-accessible user stories you created/your trello board- with given, when, then statements for each story.**
- A link on your portfolio site to your project 2.

Project #2: Building Your First Full-stack Application

Overview

This second project is your first foray into **building a full-stack server application.** You will be **building a Node/Express App with Mongo/Mongoose,** which means that you will learn about what it takes to build a functional application from the ground up yourself.

This is exciting! It is a lot, but we have given you the tools over the last few weeks to build what you need; you get to be creative in choosing what sort of application you want to build!

You will be working individually for this project, and you will be designing the app yourself. We hope you will exercise creativity on this project, sketch some wireframes before you start, and write user stories to define what your users will want to do with the app.

You will need to run your user stories, wireframes, database design, and an app deployed on Heroku by your instructors to get their feedback and approval before you begin coding! Remember to keep things small and focus on mastering the fundamentals – **scope creep/feature creep** is the biggest pitfall for any project!

Identify what you need to build and accomplish to meet project expectations and identify everything else as stretch goals. If you meet your MVP (Minimum Viable Product) ahead of schedule, you can decide which reach goals to focus on for the remainder of your time.

Technical Requirements

Your app must:

- **Have at *least* 3 models** (more if they make sense) – one representing someone using your application (a user), and two others that represent the main functional idea for your app
    - Remember that many-to-many relationships are difficult in MongoDB and focus on relationships being one-to-many for your MVP.

- **Have complete RESTful routes** for at least one of your resources with GET, POST, PUT, and DELETE

- **Write Mongo queries using the Mongoose module** and interact with your document database. Promises will make your life easier as you start to build complex queries!

- **Include wireframes** that you designed during the planning process

- **Include User Stories**

- **Include ERDs**

- Have **semantically clean HTML and CSS**

- **Be deployed online** and accessible to the public via Heroku

---

Necessary Deliverables

- A **working full-stack application, built by you**, hosted on Heroku
- A **link to your hosted working app** in the URL section of your Github repo
- A **git repository hosted on Github**, with a link to your hosted project, and frequent commits dating back to the **very beginning** of the project. Commit early, commit often. Make your commit messages descriptive.
    - Your git repo should have **at least** 30 commits
- **A `readme.md` file** with explanations of the technologies used, the approach was taken, unsolved problems, and notes to yourself so you can come back to your project later in the course and be able to pick up your train of thought, etc.
- **Wireframes of your app**, hosted somewhere & linked in your readme
- A link in your `readme.md` to the **PUBLICLY-accessible user stories you created/your trello board- with given, when, then statements for each story.**
- A link on your portfolio site to your project 2.

Submit your project to Schoology due date indicated on the class schedule!

Suggested Ways to Get Started

- **Begin with the end in mind.** Know where you want to go by planning with wireframes & user stories, so you don't waste time building things you don't need
- **Don’t hesitate to write throwaway code to solve short-term problems**
- **Read the docs for whatever technologies you use.** Most of the time, there is a tutorial that you can follow, but not always, and learning to read documentation is crucial to your success as a developer
- **Commit early, commit often.** Don’t be afraid to break something because you can always go back in time to a previous version.
- **User stories define what a specific type of user wants to accomplish with your application**. It's tempting to just make them *todo lists* for what needs to get done, but if you keep them small & focused on what a user cares about from their perspective, it'll help you know what to build
- **Write pseudocode before you write actual code.** Thinking through the logic of something helps.

Potential Project Ideas

Cheerups

The world is a depressing place.

Your task is to create an app that will allow people to create and share "cheerups" - happy little quips to brighten other peoples' days. Cheerups will be small - limited to 139 characters. Members will be able to promote Cheerups that they like and maybe even boost the reputation of the Cheerupper.

Bookmarklet

You will create an application where users can bookmark links they want to keep.

But what if users could trade bookmarks for other bookmarks? Or sell bookmarks for points? Or send bookmarks to your friends. Or something even crazier.

Closet/Bookshelf/Pantry Organizer

An application that helps users keep track of items in their closet.

Maybe users could loan books/clothes/pantry items to friends and keep track of it in their database.

Project 2 Project & Readme Samples

- https://github.com/chris-mears/WDI_Project_2
- https://github.com/jroyals2/Project-2-Brew-Tap
- https://github.com/tombeau19/backcountry_ski_app
- https://github.com/jwats287/portfolio-creator

Github `readme.md` files are written in a language called Markdown which is compiled into HTML. You'll be using this a lot so it's good to learn some simple syntax rules now.

- https://guides.github.com/features/mastering-markdown/
- http://www.markdowntutorial.com/

Project 2 Trello Board Example

Sample Project 2 User Stories

Useful Resources

- **Heroku** *(for hosting your back-end)*
- **Writing Good User Stories** *(for a few user story tips)*
- **Presenting Information Architecture** *(for more insight into wireframing)*

Project Feedback + Evaluation

- **Project Workflow**: Did you complete the user stories, wireframes, task tracking, and/or ERDs, as specified above? Did you use source control as expected for the phase of the program you're in (detailed above)?
- **Technical Requirements**: Did you deliver a project that met all the technical requirements? Given what the class has covered so far, did you build something that was reasonably complex?
- **Creativity**: Did you added a personal spin or creative element into your project submission? Did you deliver something of value to the end user (not just a login button and an index page)?
- **Code Quality**: Did you follow code style guidance and best practices covered in class, such as spacing, modularity, and semantic naming? Did you comment your code as your instructors as we have in class?
- **Deployment and Functionality**: Is your application deployed and functional at a public URL? Is your application free of errors and incomplete functionality?
- **Total**: Your instructors will give you a total score on your project between:

Score	Expectations
0	*Incomplete.*
1	*Does not meet expectations.*
2	*Meets expectactions, good job!*
3	*Exceeds expectations, you wonderful creature, you!*

This will serve as a helpful overall gauge of whether you met the project goals, but **the more important scores are the individual ones** above, which can help you identify where to focus your efforts for the next project!

Project Week

Attendance

You are required to be present by 10:00 am EDT each day during the project.

Stand-ups

We encourage you to continue having student-run stand-ups each morning. During your standup, emphasize on answering the following questions:

- What did I work on yesterday
- What am i trying to get done today
- What is preventing me from getting this done.

This meeting should take no longer than 15 minutes.

If you have ideas on how you can help a fellow-student with work that they are stuck on, please follow-up with information AFTER the stand up.

Where to go for help during project week

1. Seek out help online
2. Seek out help with your classmates
3. Seek out the help of instructors and TAs

How to ask for help

1. WHAT YOU ARE TRYING TO SOLVE:

- Be able to give a thorough explanation of the feature or user story you're working on.

2. DETAILED DESCRIPTION OF THE BUG/ERROR:

- A detailed description of the problem, the bug, and/or the error. This means: the full steps to reproduce and references to the line number of where the relavent code is.

3. WHAT I'VE TRIED

- List everything you've done to solve the bug on your own in detail. List all resources you've looked up and tried to implement and provide links.

4. QUESTION

- After going through all this what is your questions specifically, more specifically than how can I make this work?

Plagiarism Review

Statement

General Assembly takes academic honesty very seriously, and as such will not tolerate any student who plagiarizes in order to satisfy class requirements. Programs at General Assembly are intensive, require a lot of work on the part of the student, and students will occasionally not be able to complete work in a timely fashion.

Rather than rely on work that's not your own to create the appearance of success, let your instructional team know as early as possible that you are not prepared for the work. It is much easier to come up with a plan, than it is to succeed if you've been removed from the program.

Definition

Plagiarism is the act of claiming that work that does not belong to you is in fact your own. It can take many forms, and each concentration at General Assembly will have a slightly different guideline to identify plagiarism in their field.

Generally it is safe to assume that if you are including work that does not belong to you in an effort to build on your own work — cite it. If you are using work that does not belong to you as a replacement for your own work, you're probably plagiarizing.

Plagiarism can unintentionally occur. While it is fine to use outside sources, if **all** or **most** of your code (specifically logic/JS/Ruby) is from other git repos/videos/blog posts, you are in violation of the plagiarism rules cited above. If you do use outside resources, make sure you **cite** those resources in your README.md file.

Procedure

If an instructor is in doubt of a students work or has evidence of plagiarism, the student will be asked to justify the work they've submitted. This often is in the context of making you explain your code line by line. Should they be unable to show the work they have claimed as their own is in fact theirs, the student will face disciplinary action.

Instances of plagiarism will be evaluated on a case by case basis, but will most likely result in removal from the program