**TimeTrackerV2 Handoff (2022)**

**NULL GROUP**

Handoff Documentation

for

**Time-Tracker**

***Overview:*** *The primary use case for this application is to allow users to track their hours spent on projects. The application has three types of users: Admins, Instructors, and Users.*

* *Admins should be able to manually view Users and Delete Users from the System*
* *Instructors can create courses and projects, and assign users to projects. They can create and assign evals, and see the progress/statistics of each group/user.*
* *Users can join courses, view the details/projects of those courses. A user can track their time spent on a particular project by clicking the Start/Stop buttons, or by entering the time manually (some rules need to be enforced to avoid fabricated entries).*

# Our Stack:

This project is running an Angular 12 front end with a NodeJS backend paired with Express. The database is run using SQLite 3.

# Prerequisites:

These are the prerequisites for the project. It may run without some of them, but you’ll want to make sure you have each of these installed regardless. It will help you as you develop further.

VSCode: <https://code.visualstudio.com/download>

SQLite: <https://www.sqlite.org/download.html>

NodeJS: <https://nodejs.org/en/download/>

Docker: <https://www.docker.com/products/docker-desktop>

You must also ensure that WSL 2 is enabled. For more information on how to do that, look [here](https://www.omgubuntu.co.uk/how-to-install-wsl2-on-windows-10).

You can clone the repo here: <https://github.com/bradleypeterson/TimeTrackerV2>

# Startup:

For first time startup, navigate to the directory containing the docker-compose.yml (this should be the root) and run the following command:

docker-compose up --build

For any subsequent startups, run

docker-compose up -d

The -d parameter will remove trailing logs, so omit it if you'd like logs to appear in the console. Only run with --build if you make changes to the package.json or config files on either project.

The front end should be visible by navigating to the url, “<http://localhost:4200/>”, and the backend will be running on “<http://localhost:8080/>”. You can verify the backend is running by checking that it displays “hello world” to the page when you navigate to the respective URL.

# Current State:

This project is done with a SEAN stack utilizing a sqllite database along with a node.js backend using an express server. The Project has a front end that utilized Angular 12 and bootstrap. These are a list of completed items that were accomplished during the semester of FALL 2022

Completed Items:

* Bootstrap has been integrated into the project
  + Login page is styled nicely
  + Register page is styled nicely
  + Dashboard page is styled nicely
* Login functionality exists (using hashing and salts for the users password)
* Registration is functional (for student and Instructors)
  + Admins should be hard coded into the database
* Appropriate navigation bar links are displayed according to the User type account
* There are tables in the database for:
  + User
    - Used to store user information.
    - The type column in the table is used to distinguish between student, instructor and admin accounts.
  + TimeCard
    - This is used by student users to keep track of time that is worked on a project.
    - Each entry serves as a Time Log, that has a students start time, end time and a description to enter what was worked on during that time entry.
  + Course
    - This table is used to store course information. An entry for this table is created when an instructor adds a new course, through the dashboard.
  + Course\_User
    - This table serves as to connect student users to courses. An entry for this table is created when a student registered for a course through the add course tab on a students navigation view. This table references the students UserID and the courses CourseID
  + Project
    - This table is used to create projects for a course. An entry for this table is created when an instructor creates a project for a course.
  + Project\_User
    - This table is used to connect a number of student users to a project. An entry for this table should be created by an instructor adding students to a project. When a student is assigned to a project by an instructor, a clickable card should appear on a students dashboard page that will navigate to the appropriate project page.
      * (NOTE Functionality of having instructors assign students to a Project is not yet completed. Entries for this should be hard coded into the database to view the clickable links on a students dashboard until implementation is complete)
* On the Project page, the Start Time and End Time buttons successfully start and stop a time entry, timetracked for each member of the project is displayed in a history log and is tracked in a pie chart
  + Time logs are successfully pushed to the database
* An Instructor can create a new course through their dashboard and those courses appear as clickable links on their dashboard
  + New courses are successfully pushed to the database
* Once an instructor clicks on a course card, they will be taken to a course page
  + New projects can be created from this page, and will show up in the table
* A student user can register to courses through the add course button in the nav bar and can drop courses they are registered for
  + Courses\_Users are successfully pushed and deleted from the database
* Admin can view all users that exist in the system though the Users tab
  + An admin has the ability to delete users from the system. This deletes their database entry.
* Project\_Users where a students UserID is are hard coded into the database appear as clickable links on a students dashboard which navigates to that project page.
* All group members are displayed on a group project page as well as their time log history, only the logged in student can add time logs though that page.

Outstanding Items:

* Reports page for individual user needs to be implemented showing time logs, and comments for each time log entry unique to the User (Accessible to project members and to instructor)
* Functionality for instructors to be able to add students to a course when or after creating a Project.
  + Students should show up in table on course page
* Currently there is a search bar in the add courses page that should allow for the student to search for specific courses that are not yet functional.
* Enable evals
  + Allow instructors to create and assign evals to course users. Allow course users to fill out individual evals for themselves and for their project member
  + NOTE: There exists empty components in the repo for the future implementation of EVALS.
* Search functionality on course page should be implemented for projects and students
* Project page from instructor side needs to be implemented
* Courses and Projects need to have delete and edit functionalities

# Tips and General Advice:

* Since there is no built in debugger for this project, utilize console printing
  + Use browser console logs for debugging purposes
  + You can use the Docker container console to see console messages for the backend (NodeJS)
  + Use angular chrome extension for convenience
* **Don’t be afraid to jump in if you have no experience with Angular/NodeJS** (most of us didn’t). Here are a few links to help you learn:
  + Straight from the source: <https://angular.io/>
  + Pluralsight: <https://app.pluralsight.com/library/courses/angular-2-getting-started-update/table-of-contents>
  + How it is setup with docker: <https://medium.com/bb-tutorials-and-thoughts/dockerizing-angular-app-with-nodejs-backend-85e9d332335d>
  + Example Angular/NodeJS project: <https://medium.com/bb-tutorials-and-thoughts/how-to-develop-and-build-angular-app-with-nodejs-e24c40444421>
  + Google is your friend
* Look into live reloading. Not all of us could get it to work, but if you can, it will greatly improve your life. Without it, you will have to restart your docker container after every change.
* On the html pages, a lot of the old code was left in, but commented out if not useable/implemented
* All endpoints, database connecting/querying are written in the server.js file
* All database table entries and seeding are in the seed.js file