Gnomon Time Tracker

CS 360 — Final Project Report
Curtis Wigington and Ken Reese

https://gnomon.me

Description
The purpose of Gnomon is to provide a simple, open-source time tracker for individuals and small businesses. A key goal was to have a clean user-experience that made time-tracking intuitive and pleasurable experience. The paragraphs below will explain the specific features that we implemented.

Finished Functionality

- Clock in clock out
- Editable timesheets
- Organizations
- Export as JSON
- Prevent time exceptions
- Search users
- Responsive for mobile

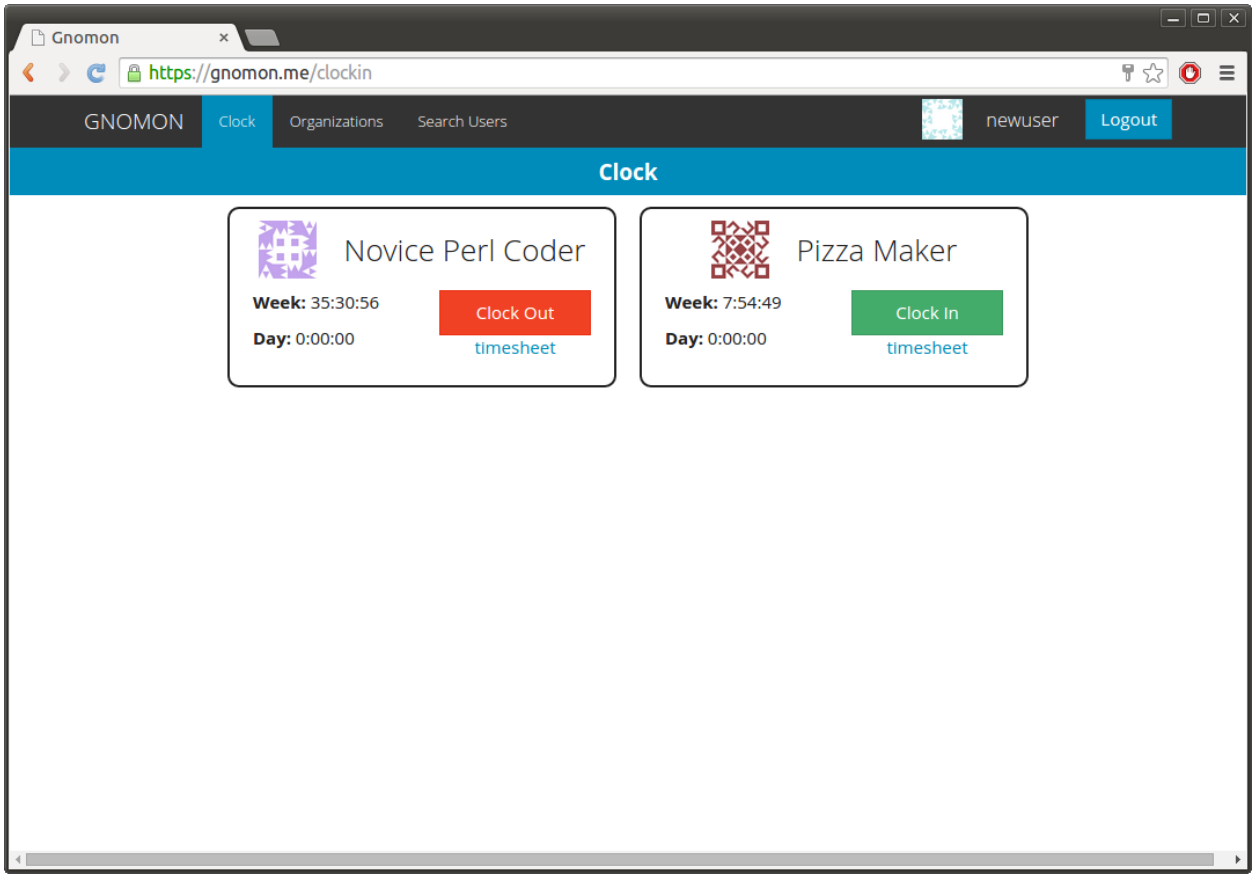


Figure 1 — Clock-in/Clock-out.

This screen allows you to easily clock in and out of jobs. Each job is identified with by a unique identicon. This page gives you day and week totals for each job, as well as a link to the timesheet pertaining to each job.

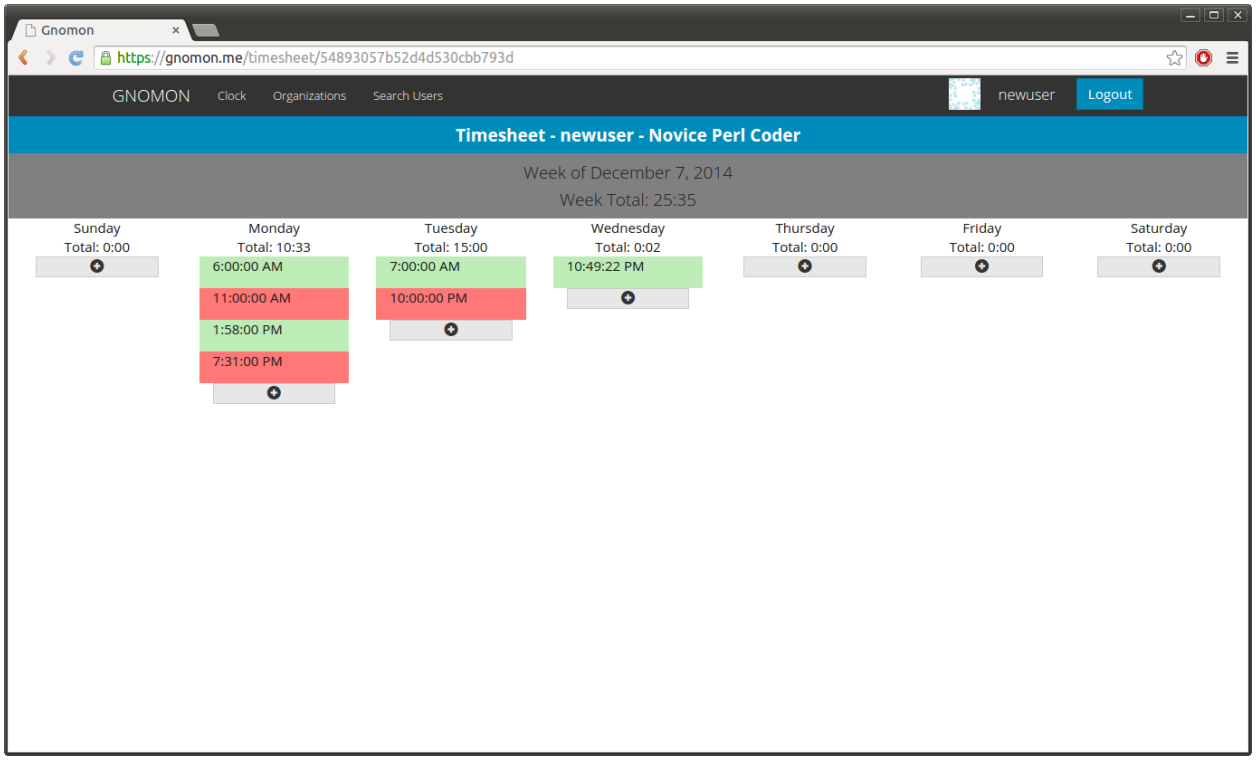


Figure 2 — Timesheet View

This screen allows you to add, insert, and delete timestamps from your timesheet. The API prevents you from making illegal times such as in the future.

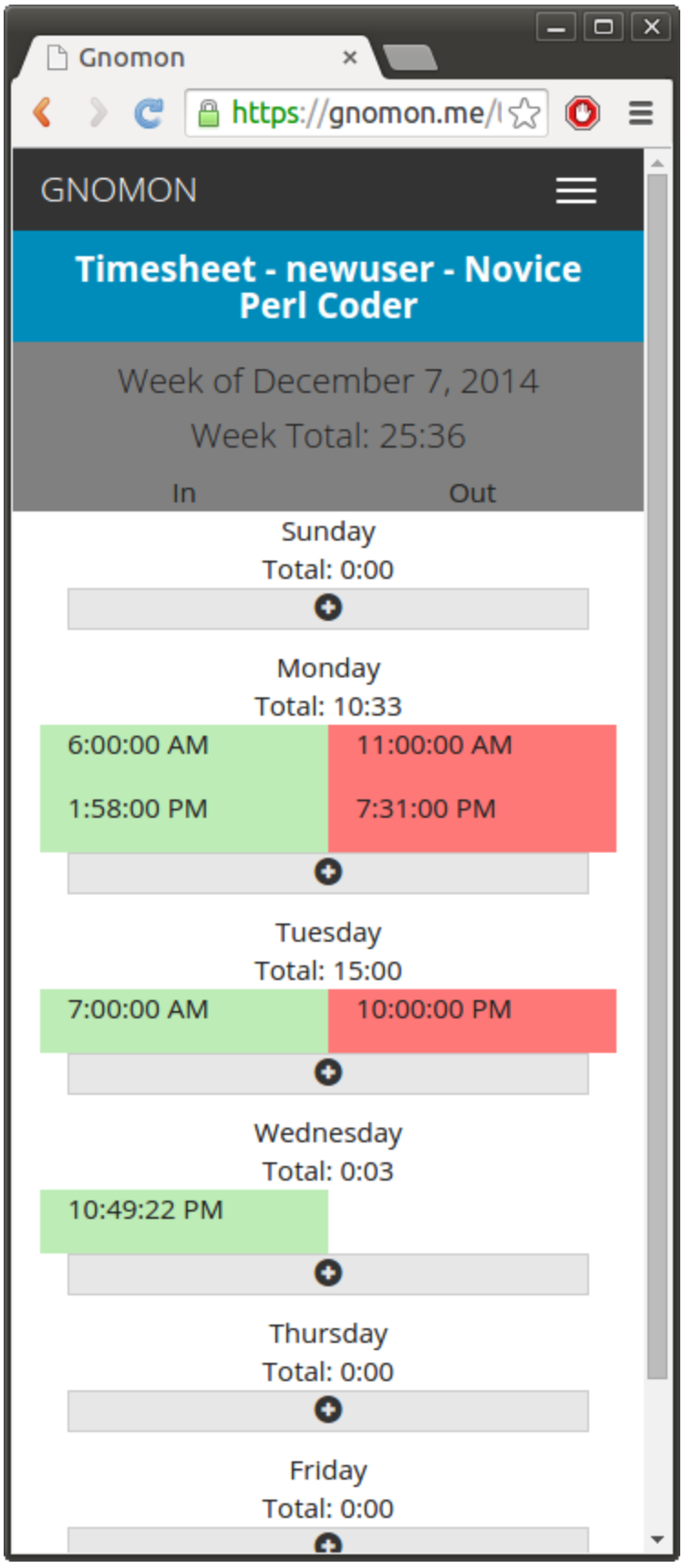


Figure 3 — Mobile Timesheet
Make edits to your timesheet from your mobile device as well. This adjust from a 7 row timesheet to a 2 row timesheet.

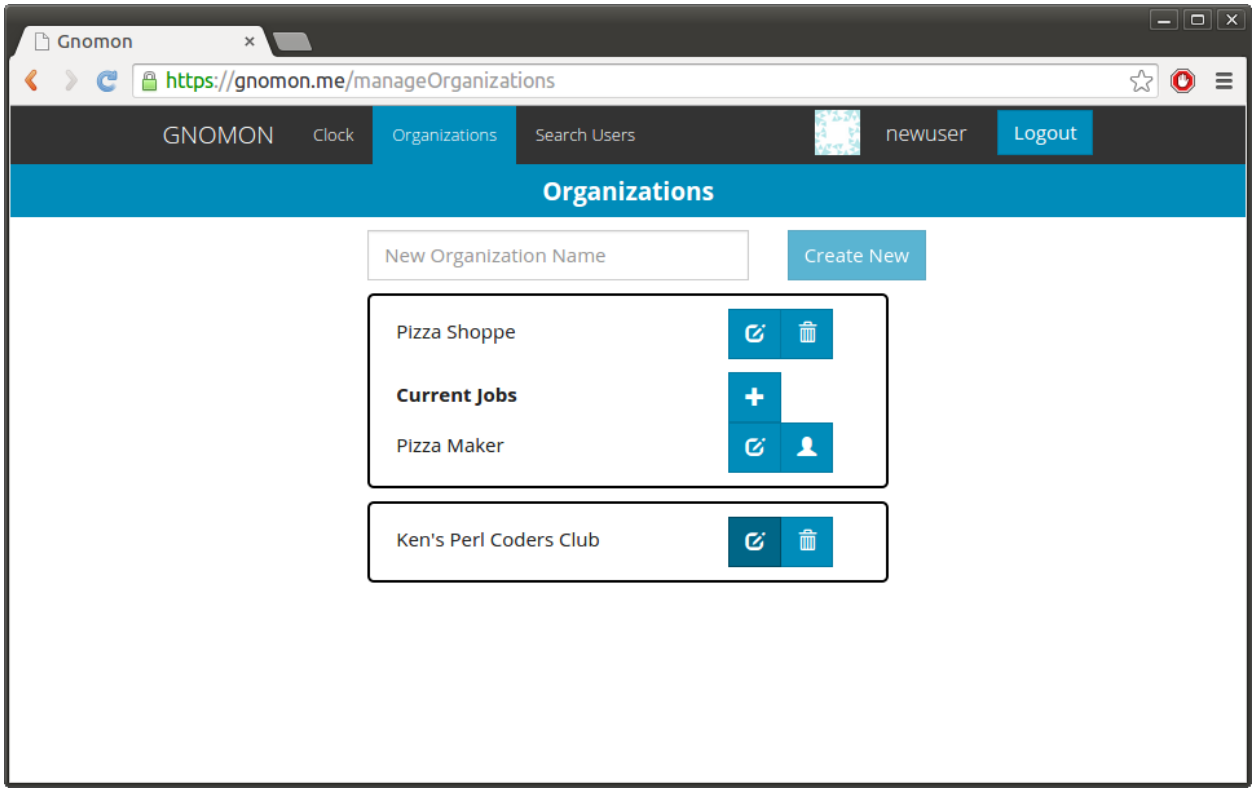


Figure 4 - Shows that you can create and list organizations. These organizations each can have a multiple jobs. You can view the users in each of your jobs. If you are the owner of a an organization you can manage the timesheets for your jobs.

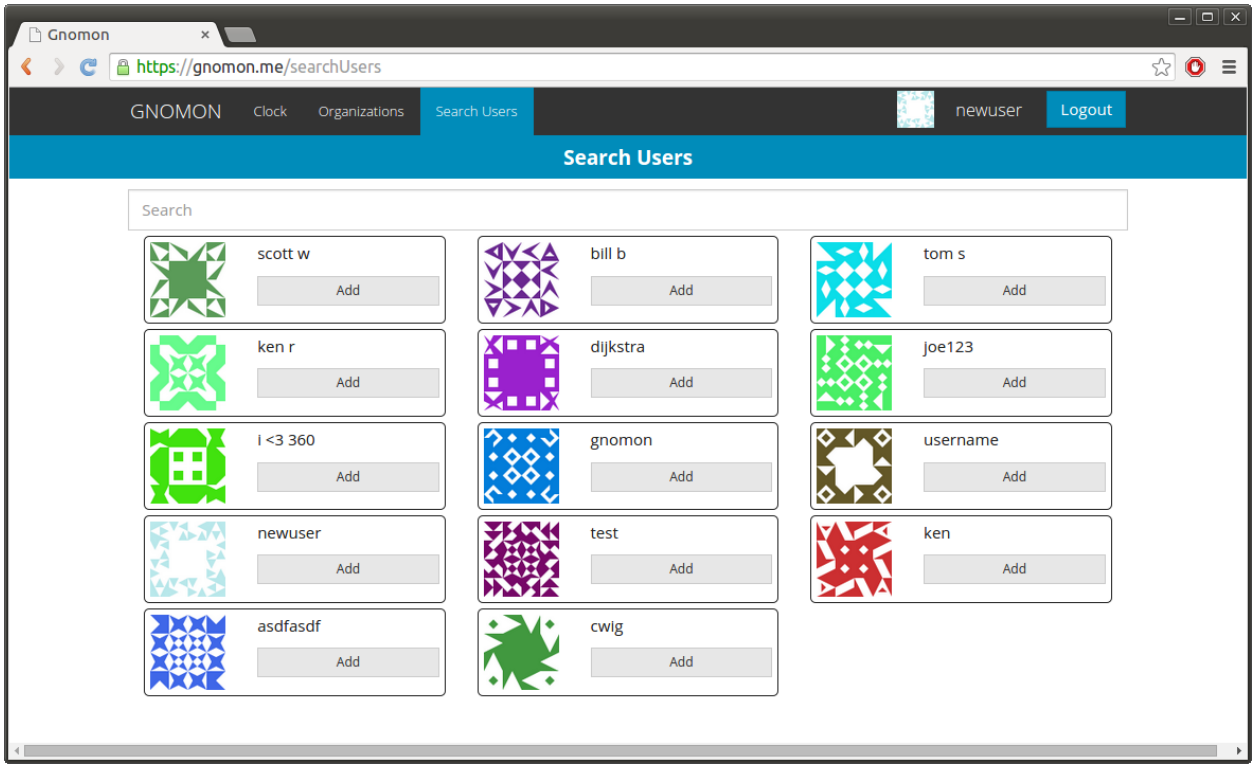
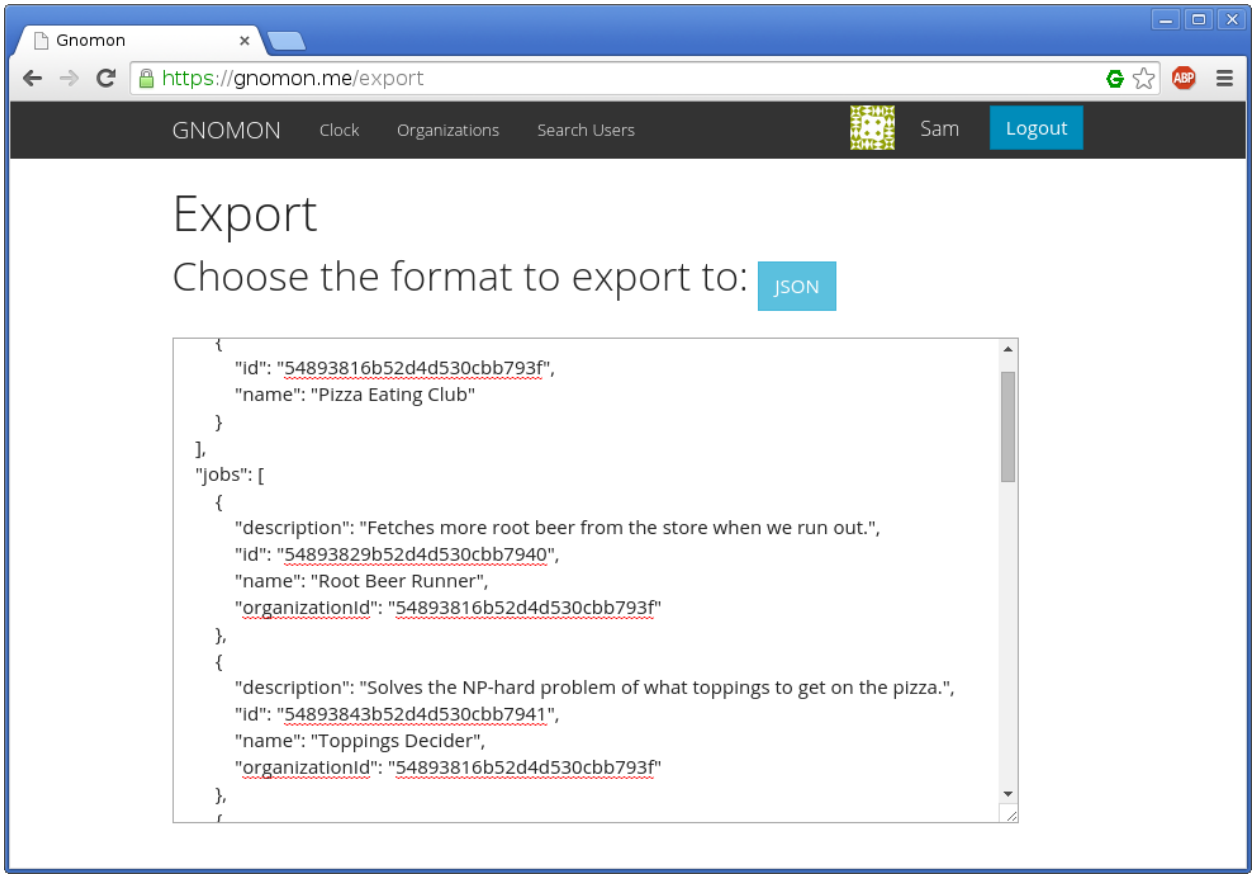


Figure 5 - In the list of users you can search for people to add to jobs. The search bar at the top will match user names so that you can find who you are looking for more easily.



Database Schema

For our database we used MongoDB. We did this because we thought that MongoDB would be good because it would allow us to make changes very rapidly. In retrospect, we are storing a lot of relational data and this turned out make things harder than if we had just used a relational database.

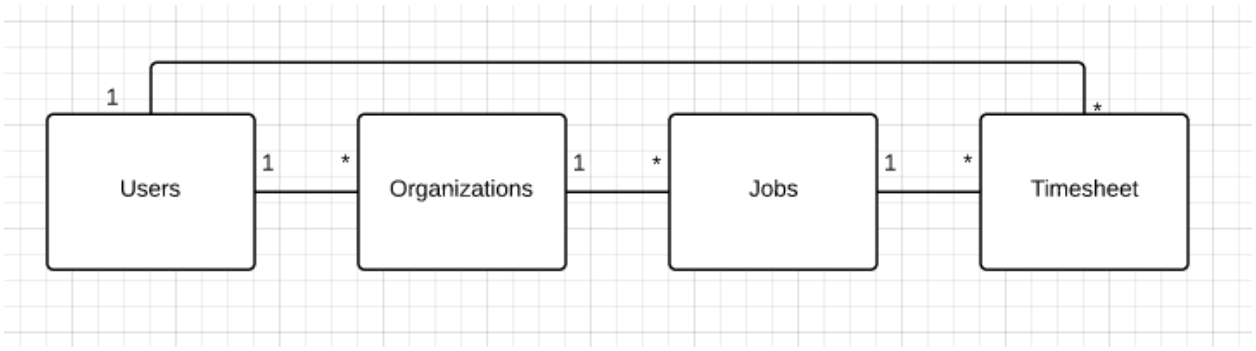
Users
{
"id": alphanumeric,
"username":string,
"created_date": ISODate,
"password_salt":string, (randomly generated for each user)
"password": hashed string (including salt)
}

Organizations
{

"id": alphanumeric,
"ownerid": alphanumeric,
"name": string
}

Jobs
{
"id": alphanumeric,
"orgId": alphanumeric,
"name": string,
"description": string
}

Timesheets
{
"id": alphanumeric,
"status": "active" "disabled" or "pending",
"userId": alphanumeric,
"jobId": alphanumeric,
"clockIn":[unixtimestamp, unixtimestamp, ...],
"clockOut":[unixtimestamp, unixtimestamp, ...]
}



As shown from the diagram above a user can have many **Users** and many **Organizations**. Within each organization there are many **Jobs.** Each **Job** has many **Timesheets**. This structure to have access to **Timesheets** for **Organizations** that own and the **Jobs** that they are in.

Future Work

Because this was a first prototype, we chose to use MongoDB, a document database. In later versions, we would definitely want to use a traditional relational database instead, since all of our data was very relational in nature. In retrospect, using a relational database probably would have been easier to implement, even at the expense of having to design a decent schema up front.

We did not get all of features completed that we think this program is capable of providing. We would like to provide the user with statistics about their hours for their organizations and their timesheets.

It would be useful for multiple user to be able to administer an organization. This would allow it to be more useful for small companies.

Our API supports adding adding a clock-in time for any week in the past. Even before the account was created. However, our GUI does not support this. We would want to work out an effective way to allow the user to do this through the site.