ManageHelp

Sprint 2† Retrospective

Team 24

Tom Carsello
Matt Hiatt
Sharan Sivakumar
Jon Tokad

What went well during Sprint 2?

We are much better at using the MERN stack now then we were during sprint 1. This meant we were able to develop better, less buggy functionality than we were in the first few weeks of the project. We added a much larger number of features this sprint than at the start of the project. Our application has really started to take shape as now there is functionality for Employees, Managers, and Administrators so the app is usable for all user groups. We have a great platform to finish off a quality piece of software during sprint 3.

Completed

User Story (4/12): *As an employee, I would like to submit requests for certain days off.*

#	Description	Time	Owner
1	Create UI for a user to be able to submit requests for days of to the manager for approval	2hr	Jon
2	Update requests for days off in the database	2hr	Jon
3	Notify users when a manager approves their request for a day off	2hr	Jon

Acceptance Criteria

- The user should have a clean and intuitive user interface to be able to submit requests for days off to the manager for approval
- Given a request for a day off is made, the appropriate manager should be notified by email and able to approve it on their approval page
- When a user's day off is approved/rejected, they should be notified by email

Comments:

The employee dashboard contains a function for submitting a request for a day off. Users input the day requested and reason and click the submit button. When they submit the request, as long as there is no error (duplicate request, lack of internet connection, etc), the request is added to their open request docket and sent to the managers. The managers can view open requests in the workspace and approve/reject them. When they approve/reject the status of the request is updated on the user's end and the user receives an email.

User Story (5/12): *As a manager, I would like to be able to approve and reject employee day off requests.*

#	Description	Time	Owner
1	Create the UI for manager to be able to approve and reject employee day off requests	4	Sharan
2	Update requests for days off in the database	3	Jon
3	Notify the appropriate user when their request is either accepted or rejected	1	Sharan
4	Notify managers and admin when a day off request is created	1	Tom

Acceptance Criteria

- Given that the user is a manager, the manager should be able to view day off requests per employee
- Given that the user is a manager, the user should be able to accept or deny requests for day offs for any given user
- Given that the user is a manger, when the user accepts or deny any given request for any given day off, the system should notify the appropriate user

Comments:

If a user is logged in as an admin or a manager, they have access to the manager functions component. The manager function component contains the option to view all open requests for days off in the workspace. The manager can approve/reject the shown ID number with the approve/reject buttons. The status of the day off request is then updated and an email is sent to the user. When a user creates a new request as described in user story 4 the relevant managers are emailed.

User Story (6/12): *As an employee, I would like to be able to request other employees cover a shift and accept other employees covered shifts.*

#	Description	Time	Owner
1	Create shift request off form	3	Jon
2	Update requests for other employees to cover a particular shifts in the database	3	Jon

3	Notify the appropriate user when a request is submitted and notify the appropriate user when their shift request off is accepted by another user	2	Sharan

Acceptance Criteria

- Given that the user is an employee, the user should be able to view all the shifts that they have for a week
- Given that the user is an employee, the user should have the option to request other employees to cover a shift
- Given that the user is an employee, the user should have the option to cover a shift that has been requested off by another employee

Comments:

Under the employee functions component, an employee can submit a request to trade shifts for a certain day. After they submit, they show as an outgoing request in the employee view function. Other employees in the workspace can then view incoming requests and choose to accept them. Once accepted by another employee, their status is upgraded to “pending with manager approval” and the posting user is notified via email.

User Story (7/12): *As a manager, I would like to be able to veto employee shift trades/coverages.*

#	**Description**	**Time**	**Owner**
1	Create shift coverage view page for managers	3	Tom
2	Create functionality to veto a shift coverage	3	Tom
3	Notify employees when their shift coverage has been vetoed	1	Tom

Acceptance Criteria

- Given that a user is a manager, the user should be able to see all shift coverages that have been made by employees
- Given that a user is a manager, the user should be able to veto a shift coverage that they do not want to be in place
- Given that a user is an employee, when their shift coverage gets vetoed, they should receive an email notification

Comments:

The manager functions component contains an option to view all current shift cover requests for the workspace. The requests display in cards with the id, status, names of both the requesting and accepting employees, date, and reason. The manager can then choose to accept or reject the request by clicking the appropriate button. When the request is accepted or rejected the requesting user is notified via email.

User Story (8/12): *As a manager, I would like to be able to view expected labor costs for a day or week.*

#	**Description**	**Time**	**Owner**
1	Create a UI for a manager to be able to view expected labor costs per day or per week	2	Sharan
2	Calculate the labor cost per day or per week	5	Sharan
3	Add labor costs as a field in the database	4	Jon
4	Update labor costs as needed in the database	2	Jon

Acceptance Criteria

- Given that the user is a manger, the user should be able to view the expected labor costs for any given day
- Given that the user is a manager, the user should be able to view the expected labor costs for a given week
- Given that the user is a manager, the user should be able to see the average cost per labor hour of a given day or week

Comments:

We did implement this feature, however we only implemented the weekly values for the time being since we failed to implement a proper schedule UI. Without a schedule UI there isn't really a logical place to input the daily breakdown, so we built the backend computations but AC1 and half of AC3 don't actually display to the user yet.

User Story (9/12): *As an employee, I would like to be able to receive notifications when my schedule is updated.*

#	**Description**	**Time**	**Owner**
1	Send emails upon creation of schedule to employees in the schedule	1	Tom

2	Send emails to specific users whose shifts have changed when a schedule is updated	3	Tom
3	Send emails to managers and admins of a workspace whenever there is an edit to a schedule	2	Tom

Acceptance Criteria

- Given that a user is an employee, manager, or admin, the user should receive an email when a future schedule has been created
- Given that a user is an employee, the user should receive an email when their shifts are changed in an existing schedule
- Given that a user is a manager or admin, the user should receive an email when any change is made to a schedule

Comments:

Despite failing to implement most of the schedule UI, we did create forms and backend for creation and editing of schedule objects. This way we were able to attach the emailing notification functionality to the backend functions for edits and creation so we could indeed complete this user story.

User Story (12/12): *As a user, I would like a way to easily navigate between different workspaces if I work two jobs.*

#	**Description**	**Time**	**Owner**
1	Create a landing page UI that displays all of the users workspaces	3h	Sharan
2	Make the displayed workspaces components route to the appropriate workspace page	2h	Sharan
3	Get the user's role for the workspace they clicked on from the database and display the correct functionality for their role (admin, manager, employee)	2h	Sharan
4	Write and deploy unit tests for routing functionality and correct role display	3h	Sharan

Acceptance Criterion

- Given that a user logs in, they should be routed to their homepage that shows any workspaces they are in along with the options to join or create workspaces.

- Given that a user is a member of at least one workspace, when they click the homepage workspace component they should be brought to the proper workspace page.
- Given that a user is properly routed to the workspace page, the page displays only the functionality they are entitled to see based on their role (Ex. Admins/managers can edit the schedule, employees can view the schedule, admins can promote employees, etc.)

Comments:

This user story worked out very well, we simply designed the home page so that you arrive at a workspace by clicking on its card that is displayed in a list of all workspaces you belong to. So all you have to do to switch between workspaces is click the homepage button in the top of every screen then select the proper workspace. Once they log in to a workspace the view the appropriate amount of functionality for their permission level (i.e. admins have all functionality, managers lack admin functionality, and employees lack admin and manager functionality.)

Not Completed

User Story (1/12): *As a manager, I would like to be able to input my company's weekly hours into the schedule template.*

Task #	Description	Time	Owner
1	Create a weekly Calendar UI	8h	Matt
2	Allow shifts to be scheduled by dragging and dropping shift elements in the calendar	4h	Matt
3	Managers should be able to assign an employee to the shift	2h	Matt
4	Add a publish button	1h	Matt
5	Add a save button.	1h	Matt
6	Write and deploy unit tests to ensure scheduling functions for managers/admin work.	4h	Sharan

Acceptance Criteria

- Given that a user is on the workspace page and is an admin or manager, they should have the option to edit the schedules.
- Given that a manager edits the schedule for a given week, they should be able to graphically view schedules.

- Given that a manager is done editing, they should be able to save their work so other managers can continue to edit it. If they are an admin they should have the additional option to publish the schedule for employees to view.
- Given that a user is not a manager, they should not be able to view or edit unpublished schedules

Comments:

We completed about half of the acceptance criteria/work for this user story. We failed to implement a proper view and edit function but we did include the functionality for saving work and prevented improper users from accessing unfinished schedules. We will have to spend sprint 3 catching up on this as this is a very important function for our application to be considered finished.

User Story (2/12): *As a manager, I would like to be able to create weekly schedules several weeks in advance.*

#	Description	Time	Owner
1	Add right and left arrow buttons to edit screen	2h	Matt
2	Correctly display dates for where the manager has navigated to via the arrows	2h	Matt
3	Create a dictionary of dates and schedule objects in the database, and attach it to the workspace object	3h	Matt
4	Write and deploy unit tests to ensure that dates are correct and the future schedules save properly	4h	Sharan

Acceptance Criteria

- Given that a user is a manager in the edit schedule screen, they should be able to navigate to future schedules up to at least a month in advance.
- Given that a future schedule is saved, its changes should be available to the next manager who attempts to edit it.
- Given that a future schedule is published, it should be visible by employees

Comments:

Since we weren't able to finish user story 1 we were also unable to finish this user story. Without an easy to use edit and view function we obviously were unable to move on to working on schedules in advance. This was a massive failure of our sprint and will have to be amended in sprint 3.

User Story (3/12) : *As an employee, I would like to be able to view any finished schedules several weeks in advance.*

#	**Description**	**Time**	**Owner**
1	Create schedule view UI	6h	Tom
2	Populate schedule view UI with published schedules in the workspace	2h	Tom
3	Add forward and back arrows to allow the user to be able to look ahead at any available schedules	2h	Tom
4	Write and deploy unit tests to ensure that schedules deploy properly and published future schedules can be viewed properly	4h	Sharan

Acceptance Criteria

- Given that a schedule is published by the admin, any employee in the workspace should be able to see the schedule in their workspace page.
- Given that an employee is viewing the schedule, they should be able to arrow forwards from the present week and see any schedules that have been published at least a month in advance.
- Given that a schedule is unpublished and in range of the viewable future schedules, it should show up as blank or with an unfinished label

Comments:

As in user story 2, our inability to finish user stories 1 and 2 made user story 3 impossible. Until managers can publish schedules all we were able to do was set up the view permissions to prevent access to unpublished schedules. Again, we will have to develop this function during sprint 3.

User Story (10/12): *As a manager, I would like to be able to receive overtime warnings if an employee is overscheduled.*

#	**Description**	**Time**	**Owner**
1	Have a label that appears whenever you attempt to set overtime hours	2hr	Sharan
2	Have an email sent to the employee if the manager ignores the warning.	2hrs	Sharan

3	Have the user notified of the change when they log back in	2hrs	Sharan
4	Update costs with the appropriate overtime multiplier in the database.	3hrs	Sharan

Acceptance Criteria

- The cost calculation algorithm should be updated with an option for overtime
- Users must see both a visual warning on the workplace homepage and an email warning sent to their personal account
- Managers and employees should be able to leave notes for each other if an employee is scheduled overtime or give the option to email the other.

Comments:

We were able to finish the first acceptance criteria. When an employee goes over 40 hours their hours after that are computed at 1.5 times their standard pay rate. However the warnings and notes features were not implemented so AC2 and AC3 have still not yet been completed.

User Story (11/12): *As an employee, I would like to be able to go back and view previous schedules up to a month ago.*

#	**Description**	**Time**	**Owner**
1	Create a drop down menu which gives the option for old schedules for all employees	4hr	Tom
2	Create copy paste functionality for the collection of weekly schedules	5hr	Tom
3	Create the display method on the backend	4hr	Tom

Acceptance Criteria

- Given that we can pull the old schedules from the database and onto the app, users should only see the last 4 weeks at least
- Given a user accesses an old schedule, old employees who are no longer with the company should still be visible
- Given that a user has been removed from the workspace, they should not be able to see old schedules even if they were on them

Comments:

We did not get to any of this functionality as is the case with most of the other schedule related user interface. This will have to be the focus of sprint 3 as it is a key component of our project.

What did not go well in Sprint 2?

In sprint 2 we had poor communication issues that we didn't really encounter during sprint 1. There were times where people were working on the same functions at the same time without knowing it and that led to time being wasted doing redundant work. Additionally, implementing the schedule proved to be more of a technical challenge than we thought which caused us to delay prioritizing it until it was too late. We emphasized working on features that we knew we had the skill to accomplish early on, which didn't give us enough time to learn and prototype the interactive schedule. This cost us several user stories and has changed what we have to focus on for sprint 3. Our communication and management issues also led us to have some documentation slip through the cracks. We didn't properly fill out our testing document which was part of the reason we had inadequate testing. Specific failures are listed out per user story above in the "Not Completed" Section.

How should we improve?

It is paramount for sprint 3 that we invest more time upfront in planning and communication. We need to specifically outline and stick to the planning document's description of who will be responsible for what. We also need to invest more time in testing as that was something we left off until the very end of sprint 2 and was more a hobbled together last minute strategy than an efficient and effective way to improve our performance. We need everyone to start early and maintain a high work ethic through all 3 weeks of the final sprint so that we can have a presentable project at the end that we are proud to present.