

Project Planning

Monday, November 26, 2018 4:15 PM

Todo

- Backend

- Design API/endpoints
 - Identify resources that require actions
 - Determine what HTTP requests are needed for each resource
 - Name endpoints
 - Identify return types for each endpoint and request type and each case of input (failure types, etc.)
 - Create API diagram
- Establish proper SQL server set up (as opposed to using local SQL within VS)
- Security
 - Implement OAuth API authentication
 - Implement permissions for API endpoints
 - Is this needed? For Chris' purposes, maybe not, but for scaling and extensibility, yes
- Implement API
 - Create OpenAPI specification (probably, will evaluate benefits after API is designed)
 - Create skeleton of endpoints
 - Create all models
 - Create concise list of questions for Chris to iron out model details
 - Create data layer
 - Move all data operations to data layer, out of API
 - Create services to interface data layer with API (research)
 - Implement logic where required

- Frontend

- Design views
 - Identify all required views
 - Determine what information is needed for each view
 - Elaborate on lo-fi designs (fully flesh them out)
 - Create hi-fi (mid-fi?) designs
 - Design color palettes
 - Digital designs (maybe like Discord's concept art?)
 - ◆ Start with hi-fi design of app layout (navigation bars, quick access menu, etc.)

- Testing

- Create Postman collection for all endpoints/requests
 - Create scripts to help automate endpoint testing
- Implement backend automated test suite

- General Documentation

- Create sprints and project timeline
- Create API documentation/guide
- Create formal requirements document
- Create overall paper

Sprint Planning

Friday, December 7, 2018 1:38 PM

Notes

- **Big note: Add keyboard/tab controls to everything**
 - o Highlight importance of accessibility
 - o Add highlight box/outline

Rough Sprints

Sprint	Dates	Tasks	Status
1	December 5 - December 19	Note: Fall finals, may combine with <i>Sprint 2</i> since progress may be slower - Further refine the project management component requirements for the app <ul style="list-style-type: none">• Define requirements more specifically• Determine resource requirements and actions - Meet with client <ul style="list-style-type: none">• Use this to determine PM requirements• Verify current data model fields• Request further information on unknown fields• Demonstrate basic app capabilities and discuss UI design• Account/login requirements - Ensure current requirements are fully fleshed out	Meeting with Chris concluded - Priority of features updated: <ol style="list-style-type: none">1) Workorders and time tracking associated to WOs2) Materials and project management stuff - Discussed having regular meetings in the New Year <ul style="list-style-type: none">o Coinciding with sprints (biweekly)? Opposite weeks of class check ups?<ul style="list-style-type: none">▪ Chris does not have a preference - Refined time tracking requirements - Workorder signatures and WO flow is critical - UI considerations moved to New Year <ul style="list-style-type: none">o Chris doesn't have a preference - Did not discuss: <ul style="list-style-type: none">o Accounts/permissionso Invoice requirements Note: Update sprints to reflect updated priority
2	December 19 - January 2	- Finish fleshing out API and endpoints <ul style="list-style-type: none">• Create API diagrams (endpoints, request types, responses)• Create OpenAPI specification• Implement skeleton for all endpoints• Planning stage completed - Refine lo-fi design concepts, create hi-fi designs for basic pages <ul style="list-style-type: none">• Details?• UI to present to Chris - Establish basic structure for automated testing of the services - Establish proper SQL database connection (this sprint or next)	- Christmas break, little progress - Researched SQL databases for best choices <ul style="list-style-type: none">• Leaning towards Postgres for it's JSON abilities to allow for customizable fields (i.e. for identifying fields of materials)• Prefer to keep it as a free/open-source database - Push UI design back to sprint 3/4 - Push API diagrams and design to sprint 3/4 - Push automated test set up back - Meeting with client: <ul style="list-style-type: none">• Gathered PM requirements• Updated priorities:<ol style="list-style-type: none">1) Workorders2) Time tracking & project management3) Inventory
3	January 2 - January 16	- Meet with client <ul style="list-style-type: none">• Discuss/demonstrate progress• Discuss lo-fi and hi-fi designs• Questions - Security implementation on backend <ul style="list-style-type: none">• OAuth 2.0• API permissions? Discuss with client - Begin frontend implementation for previously created hi-fi designs <ul style="list-style-type: none">• Primarily workorders• Account/authentication details if required - Build out Postman collection, create for automation - Build out workorder and associated endpoints <ul style="list-style-type: none">• Begin working on sort/filter/pagination solutions - Continue designing hi-fi designs for remaining pages (ongoing) - Continuous integration?	- Push meeting with client to sprint 4 - Push front-end to sprint 4 - Little work done due to being away for most of the sprint - Initial Project Management model and database design began, more work and finalizing needed
4	January 16 - January 30	- Build out inventory endpoints (materials and vendors) - Continue frontend implementation <ul style="list-style-type: none">• Inventory management - Research requirements for integrations <ul style="list-style-type: none">• UR Financial for billing• UR intranet or other source for workorder request creation - Ensure automated tests are created for all implemented functionality - Client and advisor meetings to discuss current progress and future needs	- Frontend work <ul style="list-style-type: none">• Navigation and routing mostly complete• PMQA roughed out, needs functionality• Began roughing out content area and tables• Basic HTTP functionalities implemented - Project Management wireframing mostly completed <ul style="list-style-type: none">• More requirement gathering and verification with client needed - Revamped future sprints to be more accurate - Began documentation for R&S and testing
5	January 30 - February 13	- Project Management <ul style="list-style-type: none">• Finish model/DB design• Create models and API• Roughly finish PMQA UI - Documentation <ul style="list-style-type: none">• Finish R&S document• Begin testing plan & execution document• Begin system & object design document• Create API diagram, update DB/model diagram, update architecture diagram - Frontend <ul style="list-style-type: none">• Iron out content area design (wireframes and UI)• Roughly finish workorders (content area, table design)• Begin calendar implementation• Convert to Bootstrap - Backend <ul style="list-style-type: none">• PM objects and API• Implement pagination & filtration (start with workorders)• Accounts and RBAC - Testing <ul style="list-style-type: none">• Design proper tests and validation• Begin test implementations on both backend and frontend	- Met with Chris: <ul style="list-style-type: none">• Identified requirements for accounts/permissions• Further refined status/workflow system• Discussed new required features within workorders• Discussed estimate tooling and material order/price history - R&S document completed <ul style="list-style-type: none">• Submitted to Urcourses - Frontend work <ul style="list-style-type: none">• Designs completed• Workorder pages in progress• Bootstrap conversion in progress - Backend work <ul style="list-style-type: none">• Implementation of pagination & filtration started - Moved to February break: <ul style="list-style-type: none">• Backend:<ul style="list-style-type: none">o Accounts & RBACo Pagination and filteringo Finish API design for remaining componentso Implement endpoints• Frontend:<ul style="list-style-type: none">o Bootstrap conversion (try to only use CSS, no jQuery or JS)• Testing:<ul style="list-style-type: none">o Create basic automated testing for the backend

			o Configure frontend test frameworks
6	February 13 - February 27	<ul style="list-style-type: none"> - Integrations <ul style="list-style-type: none"> • Inquire about UR Financial Services • Inquire about WO input (intranet?) - Documentation <ul style="list-style-type: none"> • Finish S&O design document • Begin thinking about business plan and how-to-use documents - Frontend <ul style="list-style-type: none"> • Roughly create inventory pages • Start reporting pages - Backend <ul style="list-style-type: none"> • IMPORTANT: Implement HATEOAS • Identify reporting requirements wrt. Backend API/models - Testing <ul style="list-style-type: none"> • Add required tests based on new functionality - Meet with client and advisor - Debugging/user experience testing <ul style="list-style-type: none"> • Formal in-house UI/UX testing after client meeting thoughts 	
7	February 27 - March 13	<ul style="list-style-type: none"> - Continue implementations of any incomplete features - Implement feedback from client - Begin polishing - Documentation <ul style="list-style-type: none"> • Continue working on final documentation • Start designing poster • Start final presentation 	
8	March 13 - March 31	<p>Note: Final sprint</p> <ul style="list-style-type: none"> - Client meeting (start of sprint) <ul style="list-style-type: none"> • Give demo to client, allow him to try it out <ul style="list-style-type: none"> o Monitor usage o Take down feedback <ul style="list-style-type: none"> ▪ UI, UX - Final draft for all documents <ul style="list-style-type: none"> o All Edits, changes - Finish implementing last minute features - Finish debugging/testing <ul style="list-style-type: none"> o Finish all the final testing - Client meeting (early sprint) <ul style="list-style-type: none"> o Discuss changes made since last meeting o Last minute change requests - Practice Presentation - Final Advisor Meeting (early sprint) <ul style="list-style-type: none"> o Pre-launch notes/concerns o Run through presentation 	

Remaining

Tuesday, March 12, 2019 12:49 PM

- Project Planning:
 - ☐ Clean out issues/ZenHub and redo it all properly (utilize this, go into more details, add as we go)
 - ☐ Identify tasks that will be done for project day, roughly design sprints for the remaining tasks
- Documentation:
 - ☐ Create a how-to document for the UI
 - ☐ Create an API reference document (finish the OpenAPI file)
- Deployment:
 - ☐ Find a way to serve web app from server
 - ☐ Package server and app into service
 - ☐ Create installer
 - ☐ Determine if LocalDb is acceptable for this version of the production build
 - Otherwise, set up proper DB service and add to the installer
- General:
 - ☐ Server-side pagination
 - ☐ Server-side filtering and search
 - ☐ Fix broken update functionality (many)
 - ☐ Ensure styling is adequate and consistent
- Notifications:
 - ☐ Identify what notifications will be required
 - ☐ Design notifications API
 - ☐ Create notifications objects
 - ☐ Style notifications
- Workorders:
 - ☐ Input form for administration side
 - ☐ Input form for client side
 - ☐ Finish details page
 - ☐ Finish view all page
 - ☐ Edit workorder page
 - ☐ Add materials
 - ☐ Associate time entries
 - ☐ Finish state/workflow buttons/interactions
 - ☐ Create comments list
 - ☐ Export individual workorder
 - ☐ Determine authorization/verification signature system
- Materials:
 - ☐ Input form
 - ☐ Details page
 - ☐ Edit material page
 - ☐ Finish view all page
- Vendors:
 - ☐ Input form
 - ☐ Details page
 - ☐ Edit vendor page
 - ☐ View all page
 - ☐ Ability to add materials to vendor
- Project Management:
 - ☐ All calendar views
 - ☐ Display of calendar events/availability/time-entries/workorder planning
 - ☐ Daily time entry view for side menu
 - ☐ Create pages for viewing and editing all calendar object types
- Dashboard:
 - ☐ Identify stats for dashboard reporting
 - ☐ Create graphics and charts
- Reporting:
 - ☐ Identify full reporting needs
 - ☐ Save report as template
 - ☐ Run report template
 - ☐ Export as PDF (others?)
 - ☐ Create view all reports page
 - ☐ Create report details page
 - ☐ Create report API (unknown requirements)
- Billing:
 - ☐ Determine billing needs and billing process
 - ☐ Design billing API
- Settings:
 - ☐ Customizable workflows (many large components)
 - ☐ Identify all customizable information and add appropriate sections
 - ☐ Permissions management
 - ☐ Accounts and roles management
- Account Management:
 - ☐ SAML authentication
 - ☐ Client portal and access
 - ☐ Access permissions

Tasks that will be in a later release:

- Reporting
- Settings and customization
- Dashboard (?)
- Billing (?)
- Notifications
- SAML (will try to get basics before)

- Testing:
 - ☐ Write all Postman requests
 - ☐ Create multiple iteration requests in Postman to populate database
 - ☐ Unit tests for server
 - ☐ Set up testing

Meetings

December 7, 2018 4:51 PM

Dec. 19th, 2018 – Meeting with Chris

Questions/topics	Notes from meeting (Jon)	Notes from meeting (Kon)
<p>- Required model data</p> <ul style="list-style-type: none"> Figure out fields/objects that were left blank/vague <ul style="list-style-type: none"> Materials <ul style="list-style-type: none"> General fields for all materials <ul style="list-style-type: none"> Any fields specific to certain materials types/categories? Custom fields needed? Material types and categories <ul style="list-style-type: none"> Is this even accurate for identifying materials? Did you want to store material locations within the shop? Barcoding? Orders <ul style="list-style-type: none"> Should orders from vendors be associated to workorders? Should it handle wholesale pricing? (eg. Different price per different quantities) Workorders <ul style="list-style-type: none"> Signature requirements and authorization What sort of information for quotes/estimates? Any specific statuses you'd like to see? List current plan Attachment types? CAD file, drawings (pdf, dwg, dxf?) Time Entries and labour <ul style="list-style-type: none"> Different labour costs per hour? <ul style="list-style-type: none"> Any overrides needed? Just two costs? Students get 10 hours at \$5/hour Standard is \$60/hour Does only direct shop time count towards that? Different labour costs per type of labour? <ul style="list-style-type: none"> Are there even different labour types? Eg. Drilling, milling, etc. If so, do you record time spent in each? Is there a need for the "Misc. Shop" etc. time types? Do we need to record invoices for anything? <ul style="list-style-type: none"> What invoice information would be needed? How should financial data be stored? I'm thinking from Financial Services <ul style="list-style-type: none"> Invoice numbers? Double check current fields Get enumeration types Many repeat clients? Thinking in terms of storing data on each <p>- Usability</p> <ul style="list-style-type: none"> UI requests What actions he really wants to do <ul style="list-style-type: none"> Primarily project management/calendar/time tracking related How do you see yourself using the workorders? Account/permission requirements Time tracking <ul style="list-style-type: none"> How do you see yourself using time tracking? <ul style="list-style-type: none"> Enter start and end time? Enter hours done? Both? How fine of time types do you see yourself using? <ul style="list-style-type: none"> Shop, Office, Meeting? Maybe add maintenance, etc.? When recording meetings, do you record the client? <ul style="list-style-type: none"> If so, would you want to enter client information for future use? Would aid in searching too probably? Notice of projects <ul style="list-style-type: none"> How do you see yourself using them? 	<p>- Model Data:</p> <ul style="list-style-type: none"> Materials: <ul style="list-style-type: none"> Type: <ul style="list-style-type: none"> Round bar, angle iron Be able to add new fields <ul style="list-style-type: none"> Customizable tables basically Orders: <ul style="list-style-type: none"> Wholesale pricing - kind of <ul style="list-style-type: none"> Few vendors that he deals with Generally only one price for a material, but add ability to add additional prices Workorders: <ul style="list-style-type: none"> Digital signatures are very important Process in general is important <ul style="list-style-type: none"> Three step process: <ul style="list-style-type: none"> Fill out WO online Meeting associated with WO <ul style="list-style-type: none"> He assesses it, approves it, determines scope Time estimation is important for budget stuff Time Entries & Labour: <ul style="list-style-type: none"> Not really much reason to enter time as "Drilling", "Milling" <ul style="list-style-type: none"> Everything is billed as "Shop time", not specifically <ul style="list-style-type: none"> But it could help tremendously for planning Overrides: <ul style="list-style-type: none"> Undergrad work (work for labs can be free) "Fee change" Maybe add a section for administrator to moderate that Labour rate radio buttons, dropdown, etc. Invoices: <ul style="list-style-type: none"> Didn't talk about <p>- Usability:</p> <ul style="list-style-type: none"> UI: <ul style="list-style-type: none"> Flesh this out more before showing Pushed back until a January meeting Time Tracking & Project Management: <ul style="list-style-type: none"> 15 minute segments Min billing = 30 minutes Never bill for "soft" stuff (i.e. Meetings, material sourcing, etc.) Time entry as duration and start stop, try both <ul style="list-style-type: none"> Maker schedule vs manager schedule <ul style="list-style-type: none"> Manager = blocks of time (30 minutes, etc.) Maker = larger, more like half days, days, hours as a minimum Doesn't need to replace meeting calendar that already exists Calendar mainly for scheduling workorders <ul style="list-style-type: none"> Think simple Talk in New Year about his PM class <ul style="list-style-type: none"> Scope: develop time tracking software for billing time Dropdown - choice of time entry categories <ul style="list-style-type: none"> Associate with workorder Start timer, stop when workorder is closed <ul style="list-style-type: none"> Best if you work at your desk all day Best for Chris would be enter start and stop time Chris currently fills out biweekly time card for sick time Ability to enter "Out sick" or "Shop Closed" or whatever various reason - visual reference, he doesn't need to account for Notice of Projects: <ul style="list-style-type: none"> Capacity planning, estimate allotted time for workorders to schedule in the future Maybe don't worry too much <ul style="list-style-type: none"> So far, it's more for the students than him Accounts/Permissions: <ul style="list-style-type: none"> Didn't talk about <p>- Priorities:</p> <ul style="list-style-type: none"> Top priority: Workorders and basic time tracking association Last level: Materials and other project management 	<ul style="list-style-type: none"> Project Management Priority 1 Time tracking at the end of the day Memory system 15 minute segments Miscellaneous meetings <ul style="list-style-type: none"> Has its own 15 minute block 30 min billing blocks <ul style="list-style-type: none"> When billed work is done Meetings are recorded open endedly (not workorders) <ul style="list-style-type: none"> Not too specific as of yet Workorder meetings are part of the workorder tracking <ul style="list-style-type: none"> Considered soft work Soft work not billed Has not tried online tracking Maker schedule vs manager schedule <ul style="list-style-type: none"> Manager is a block of time <ul style="list-style-type: none"> Segmented like a day planner Maker is allotment of time <ul style="list-style-type: none"> Messured in half days and days Big blocks Need a mixture of both schedules Internal calendar (this refers to his current staff calendar, not the one in our app) <ul style="list-style-type: none"> Setting up meetings Similar to outlook Used more for meetings, not for project work Not used as a daily tracker Most useful as a calendar Simplicity is key <ul style="list-style-type: none"> Usability over cusomability For scheduling workorders Never marking in time as its happening Maximize time Choice of sections Drop menu to associate to workorder <ul style="list-style-type: none"> Track in real time Online punch-in/punch-out Login into workorder Ability to edit time blocks <ul style="list-style-type: none"> Edit, add, delete features to time blocks Time tracking for projects, does not need to allocate sick time, vaction days <ul style="list-style-type: none"> Labels for shop being closed still needed as a reference Likes Calendars style <ul style="list-style-type: none"> View by month, view by week, view by day Future Workorder planning Future jobs coming in New work is hard to guess how long the time takes Ability to add new materials and material types No reason to right now denote the specific labour type <ul style="list-style-type: none"> Future expansion Could be used for planning of which machine will be used Other override work <ul style="list-style-type: none"> Undergrad lab work is free New build could have a charge of \$30 per hour Red is Chris' favourite colour Material Inventory <ul style="list-style-type: none"> Plastics for davey's plastic Alwin for metals 2-3 competitors Digital Signatures for workorders <ul style="list-style-type: none"> 3 step process Be able to send to chris electronically Call a meeting before or after workorder <ul style="list-style-type: none"> Associated with the workorder Workorders cannot be approved without meeting with the client Materials is last priority

February 5th, 2019

Questions/Topics	Notes from meeting (Jon)	Notes from meeting (Kon)
<ul style="list-style-type: none"> - Status flow: <ul style="list-style-type: none"> • New > Scheduled > In Progress > Completed > Closed? • Cancelled/Rejected? • Waiting on pickup? • Separate column for keeping track of on schedule? <ul style="list-style-type: none"> ◦ On schedule > Warning, close > Late? - Workorder details: <ul style="list-style-type: none"> • Should we add a "Project Title" field? Not on your original, but might make the list more readable • Estimates: <ul style="list-style-type: none"> ◦ How do you do the estimates? ◦ Do you need any tooling built in for that? ◦ What information do you want displayed for estimates? - Project Management class notes? - Note: Changing over to Bootstrap, lots of styling/layout is likely to change <ul style="list-style-type: none"> • Will do that over the weekend - For Tim: <ul style="list-style-type: none"> • Regarding maintainability for his purposes, would it be better if we used a CSS library? <ul style="list-style-type: none"> ◦ Easier to maintain a similar style when adding new components ◦ Potentially easier-to-read CSS • Other maintainability notes? Any specific tooling or requirements you'd like to see? • Intranet? Where to submit from? Should the server be responsible for hosting a public-facing form? • Financial Services integration? 	<ul style="list-style-type: none"> - Accounts for Meeghan and Chris - For people with jobs, they rely on talking to Chris to know status of their job <ul style="list-style-type: none"> • Emails based off status? - Statuses: <ul style="list-style-type: none"> • Waiting on client • Waiting on materials • Add customizable workflow like Jira - Workorder comments with status changes <ul style="list-style-type: none"> • Like Jira • For Meeghan to know where stuff is at • Let Meeghan and Cathy leave comments too - Financial services: <ul style="list-style-type: none"> • Communicate through Cathy? - No quotes, just estimates (less iron-clad) - Inventory order history (prices @ dates) - Estimates: <ul style="list-style-type: none"> • Create estimate: <ul style="list-style-type: none"> ◦ Add material ◦ Add time/labour - Workorder notes for implementation details <ul style="list-style-type: none"> • Comment when workorder closed • Perhaps closed early, comments to record that - Cancelled workorder: <ul style="list-style-type: none"> • If billable time/materials already consumed, prompt to bill • If student cancels, send request to Chris so he can verify and cancel it 	<ul style="list-style-type: none"> - No involvement from deans <ul style="list-style-type: none"> • Meeghan main admin - Meeghan having the ability to see the different statuses - Scheduling as an add on - Status has small description <ul style="list-style-type: none"> • Additional info as to why a workorder has the status it has - Comments on workorder page <ul style="list-style-type: none"> • Additional comments from Meeghan/Kathy on Workorder - Program heads having access for workshop schedule in future implements - Ability for future implementation of sending receipts on materials - Never give quotes from the beginning <ul style="list-style-type: none"> • No need to recoup project time • Wont work more until client is notified • Materials are ordered even if not used in specific order - Date of price stored in the Inventory <ul style="list-style-type: none"> • History of price - Bill of inventory not included on workorder - No markup on any materials - Changes in price can change what materials are used <ul style="list-style-type: none"> • Notify user when materials are changed • Changes are noted on workorder - Certain requested materials should have a location of where to find it in the notes - If budget is too extreme, labour can then be complementary <ul style="list-style-type: none"> • Can be recorded as unbillable hours - Cancelled workorder may be billed, but not necessary <ul style="list-style-type: none"> • Non-built cancellations are not billed

NEXT

Questions/Topics	Notes from meeting (Jon)	Notes from meeting (Kon)
<ul style="list-style-type: none"> - History of estimates for workorder for audit reasons, or is it fine to just update 		
<ul style="list-style-type: none"> - Notes from Mark changes need to be made? <ul style="list-style-type: none"> - SAML (Library) <ul style="list-style-type: none"> ◦ For external providers ◦ Set our own entityId and DNS of server - CAS - Financial Services <ul style="list-style-type: none"> ◦ API integration into banner ◦ CSV (sld, amount, date) 	<ul style="list-style-type: none"> - Meetings with Mark Kon - SAML <ul style="list-style-type: none"> ◦ Library authentication - CAS - API integration into banner - CSV file provides student info, amount <ul style="list-style-type: none"> ◦ Financial services loads into banner 	

Time Management

December 7, 2018 4:57 PM

Dec. 7th, 2018 Meeting notes

- Establish what is needed for formal meeting with Chris
 - Finish questionnaire for Chris
 - Plan date for meeting
- How the project management page will look like
 - On the management page
 - Ideas include Monthly, Weekly and Daily Calendar
 - As a side bar
 - Upcoming dates
 - Not Calendar (Future Kon don't complain)
- Finishing up Sprint Planning

Presentation Requirements

January 18, 2019 9:29 AM

- Time planning
- Documentation
 - o Requirements
 - o Design
 - o Architecture
- Testing
- Discuss code reviews

Preliminary Discussion Notes

September 11, 2018 11:22 AM

- Inventory and Work-order Management System
- Global Shop
- Mobile app?
- Barcode item management? Material management as well?
 - o Mobile app for scanning barcodes? Handheld scanner? Manual?
 - o Eg. X is in Toolbox 1
- Provide estimates to students before work-order creation?
- Sign-in service?
 - o Assumedly would just be for you or other shop managers in the future
- Barcode to keep track of fixed assets/tools
 - o Integrate with campus-wide system (computer barcodes, etc.)

Required Features

- Dashboard
 - o Current work-orders
 - o Mini reports?
- Material Inventory
 - o What's stocked
 - o Vendor info
 - o Date of purchase
 - o Location in shop?
- Material List
 - o Vendor
 - o Cost of material
 - o Date of last purchase
 - o Price history if price has changed?
 - o View current/in-progress orders? Save receipts?
- Work-order Tracking
 - o All work-orders
 - o Work-order statuses
 - o Hours worked
 - o Billing
 - Integration with UR Financial Services
 - Store student profiles for quick additional work-orders and searching/sorting?
- Reporting
 - o Custom report generation
 - Eg. Hours worked per discipline
 - Work-orders completed in timeframe?
 - o Export to PDF?
 - o Save reports within app for later viewing
 - o Automatic reports? (monthly, quarterly, etc.)

Priority

- 1) Work-order tracking
- 2) Inventory tracking
- 3) Project management/time-tracking & report generation

- Record meeting minutes - do this for previous two meetings, do for all meetings (with Chris, with Konstantin)
 - o Participants
 - o Time and date
 - o Discussions
 - o Decisions
- Give Yasser access to our records
 - o Maintain meeting minutes in GitHub

Notice of Project on intranet site - due December 5th (first Friday?)

Work-order

- Authorization from project head
- After workorder is complete, its sent to engineering office (Cathy), then to Financial Services
- WO ID filed with Expense number
- Recurring WOs
- Add drawings
- Drive permissions for internal vs supervisors
- Future
 - o New numbering system
 - Previously sequential
 - New as of 2018: Year - Semester - Sequential ID
 - 2018-S1-001
 - Undergrad/Grad/Faculty
 - Who it was for (ISE/ESE)
 - Professor
 - o Track who is using the shop to adjust pricing
 - o Good searching

- Notice of Intent form to plan out rough shop use hours
- Students can fill out most of work-order on their own and submit
 - o Required fields
- Estimates are on request, not always
- Integration with intranet site for the shop
- Project Tracking is a big one
- Work-order number is filled in once it is started

Material

- \$3500 on purchasing card, after that it goes through the Engineering Office
- Ordered X, work-order used Y, have X-Y left
- Vendor contacts list

*Time Tracking

- Overhead
 - o Doesn't track meeting time, material sourcing time, data/calls, learning equipment
- Shop hours
 - o Tracks shop work time
- Report how much time is overhead vs. actual shop time
- Would like to track daily activities (what categories time is spent in)
 - o Categories: Office, Meetings, Shop?
 - Add finer categories to each
- Tag overhead time to workorders
 - o Add previous meetings, hours, etc. to new work-orders
- Think of SAP from SaskPower
 - o Work capacity
 - Future planning vs previous capacities
- Graphs
- Calendar of due dates, etc.

Requirements/User Stories

September 21, 2018 9:51 AM

Epics

1. Workorders
2. Inventory
3. Project Management
4. Report generation (part of 3)

General

- a. Run server as a service
 - i. Must run continually and be able to start itself after sleeping, etc.
 - ii. Must deliver front end
 - 1) URL will be server name + a self set name for the application instance

Workorder

- a. Workorder details
 - i. Time tracking
 - Time entries for meetings
 - Time entries for labour with labour types
 - ii. Tasks
 - Allow adding tasks to breakdown workorders and provide more accurate estimates
 - iii. Integrated billing
 - Auto response receipt
 - Integration with UR Financial Services
 - iv. Export as PDF
 - v. Email/share
- b. User submission
 - i. User distinction (user/supervisor)
 - ii. Supervisor details
 - iii. Contact info
- c. Status queues for workorders
 - i. Priority
 - ii. Sort/Search/Filter
- d. Workflows
 - i. State of the workorder
 - ii. Customizable workflows
- e. Workorder list
 - i. Statuses (in progress, cancelled, etc.)
 - Colour coding associated to statuses
 - Alerts/notifications based on status changes
 - ii. Sort/search/filter
 - Filter by semester
 - Filter by status
- f. Materials integration
 - i. Materials used
 - Vendor used for material
 - Quantity used
 - Cost per unit, total cost
- g. Estimates/quotes
 - i. Comparison to previous workorders
 - ii. Manual materials and labour estimates
 - Estimated total cost
 - Estimated time
- h. Create workorder from existing workorder
 - i. Label for recurring workorders

Note:

- With a customizable workflow, we will probably need something to store the status on a per workorder level, otherwise what happens when a status is removed from the workflow?

Inventory

- a. Searchable table of materials
 - i. Search options and filtering
 - Material type, material category
 - By vendor
 - ii. Search workorders by material used
 - iii. Export to CSV (?)
- b. Material details
 - i. Workorders used in
 - ii. Vendors
 - iii. Orders
 - iv. Quantity and unit type
 - v. Quality (?)
 - vi. Location in Shop (?)
 - Physical location
 - Barcodes

Project Management

- a. Project calendar
 - i. Plan/view workorder schedule and due dates
 - ii. Add meetings
 - iii. Daily time tracking
 - Ability to edit time entries
 - ◆ Tag to previous workorders, edit details, etc.
 - iv. Daily activity tracking
 - v. Calendar views:
 - Daily, weekly, monthly, schedule
- b. Time tracking
 - i. Enter time as a duration or as a start/stop time
 - Duration as 15 minute segments, billable time as 30 minute segments
 - ii. Choose type of time entry
 - Shop, meeting, office, management (i.e. out of office, sick, etc., perhaps needs different name)
 - iii. Associate time entry to workorder
 - iv. For billable time: rate modifier (such as discounted rates)
- c. Project management sidebar
 - i. Sidebar on every page
 - ii. Configurable (?)
 - iii. Time entries
 - Add new entries
 - View breakdown of time today (?)
 - iv. Calendar (Daily view? Weekly? Full?)
 - View scheduled workorders
 - View upcoming due dates
 - Add/view meetings
- d. Work capacity planning

- e. View workorder queues
 - i. Filter orders by current date, semester, etc.
 - ii. Colour coding
- f. View basic reports

Report Generation

- a. Create new report
 - i. Choose criteria, filter, etc.
 - ii. Schedule to run periodically (eg. Monthly, quarterly, etc.)
 - iii. Save reports to a database
- b. View reports
 - i. Export to PDF
 - ii. Share
 - iii. Create new from existing report
- c. Save report templates

Additional Feature Ideas

Monday, February 4, 2019 8:22 PM

Workorders

- Additional statuses based off workorder aspects (quote requested, etc.)
 - o Could automatically create a task and due date for creating the quote and update the status based on it
- Recommend which type of work to be done on a particular day

Settings

- Ability to change colors for status types (is this important, or is it sufficient to have green, yellow, red, grey?)

Inventory

- Link directly to Vendor?

Actions

Monday, December 3, 2018 2:32 PM

Dashboard

Workorders

- Workorders
 - o CRUD
 - o Search workorders by keyword
 - o Filter workorders by specific filters
- Workorder materials
 - o Add
- Time and labour

Inventory

- Materials:
 - o CRUD
 - o Search materials by keyword
 - o Filter materials by specific filters
 - o View all vendors of material
 - o Add new vendor for material
- Vendors:
 - o CRUD
 - o Search vendors by keyword
 - o Filter vendors by specific filters
 - o View all materials from specific vendor
 - o Add new material available from vendor

Project Management

- Time Entry:
 - o Specify time
 - Options:
 - ☐ Start & stop times
 - ☐ Start time and duration
 - o Associate to a workorder
 - Dropdown of all current/active WOs
 - o Specify time type
 - Dropdown with the three time types
 - o Specify billing rate modifier
 - Dropdown with modifiers/overrides
 - ☐ Only visible if billable time i.e. Shop time
- Calendar:

Project Management Quick Access

Reports

Capstone Requirements

January 11, 2019 8:39 AM

Capstone Requirements

- Poster
 - o 24" x 36"
 - o End of February
- Project Abstract
 - o Mid February
- Project titles
- Group members
 - o Advisors
 - o SIDs
 - o End of January
- Scheduling and practice room bookings
 - o March

Project Day Notes

- April 6th, 2019
 - o 8:30 - 16:30
 - o Be there by 8:00 (breakfast woot)
- Presentating to industry members
- Consider safety aspects of project
 - o i.e. Less materials needed for storing physical data
- Poster development session available
- Posters should present the data we would share in presentation
- Trade show
 - o SSE and ESE mostly use this
 - o Day before project day (April 5)
- Attendance is mandatory throughout the project day
- Doors are monitored

Documentation (Final docs due Apr. 8th)

- S&O - class diagram, system architecture
- TP&E - description of tools, document cases (test logs)
- Business plan - budget, revenue/savings

Project Abstract

January 11, 2019 09:44

Draft 1:

The Workshop ERP (Enterprise Resource Planning) Suite is an application that manages the administrative tasks of the engineering workshop on campus. It manages workorder forms that are submitted by faculty and students, manages the inventory of the shop, as well as tracking all kinds of projects, scaling from large scale projects to single day repair jobs. Workorders are submitted through an electronic form that can be evaluated for future meetings with the workshop manager. Inventory of the shop is stored on databases that the manager can use for tracking of all materials and the attached necessary information. The system is designed to replace the original method of tracking workorders and inventory in the shop by tracking all aspects of the shop online.

Draft 2:

The Workshop ERP (Enterprise Resource Planning) Suite is an application that manages the administrative tasks of the engineering workshop on campus. It is responsible for managing workorders, workorder capacity planning, time tracking, and the materials and parts inventories. The workorders are submitted online by students and faculty members which can then be reviewed by the shop manager. From there, the status of the workorder can be monitored and updated throughout the entire job. Time, both billable and nonbillable, can be recorded towards each workorder, as well as the parts and materials used. The overall inventory of the shop can be recorded so that the manager can track all of the materials and any related information. The system is designed to modernize the original method of tracking workorders, inventory, and time in the shop by moving it all to a powerful central system.

Draft 3:

The Workshop ERP (Enterprise Resource Planning) Suite is an application that manages the administrative tasks of the engineering workshop on campus. It is responsible for managing workorders, workorder capacity planning, time tracking, and the materials and parts inventories. The workorders are submitted online by students and faculty members which can then be reviewed by the shop manager. From there, the status of the workorder can be monitored and updated throughout the entire job. Time, both billable and nonbillable, can be recorded towards each workorder, as well as the parts and materials used. The system is designed to modernize the original method of tracking workorders, inventory, and time in the shop by moving it all to a powerful central system.

Notes

- Include the term time tracking
- Mention managing workorders through their entire life cycle, managing their status? Does that seem worth mentioning or nah?

Risks

January 11, 2019 09:20

Programming Risks

- Backend does not communicate with frontend as intended
- Programming languages not compatible
- Unresolvable bugs
- Inventory management system may be out of scope
 - Certain features may be too complicated for first release
- Designed functionality proving to be impractical/infeasible
- Different but interdependent data not compatible

Documentation Thoughts

Wednesday, December 12, 2018 5:16 PM

- GitHub Pages API documentation
 - o Allows for nice formatting
 - o Easily viewable online

R&S Notes

February 8, 2019 12:45 PM

Constraints:

- Can reword the mobility part:
 - Due to the data-heavy nature of enterprise applications, the service is not designed for viewing on mobile devices and thus there is no guarantee for mobile experiences.
- Non-technical constraints:
 - Reword restraints to constraints
 - Do we need to follow guidelines of conduct? Those are more behavioural, not really application based
 - Also don't need the understanding of computing part

System & Objects

February 18, 2019 12:29 PM

Table of Contents (rough estimate)

1. Introduction
 - a. System Goals & Objectives
 - b. Overview
 - i. System Architecture & Integrations
 - ii. Technology & Tooling
2. System
 - a. Server
 - i. General Structure (solution/project layout, what goes where)
 - ii. Authentication (account authentication, login)
 - iii. Authorization (role-based access control)
 - iv. Workflow Engine (does this need to be described here, or just show the model in the next section?)
 - b. Client (not 100% sure what this section needs to look like)
 - i. General Structure (Vue project layout?)
 - ii. Management Interface
 - iii. Client Submission Interface
 - c. API Interface (resources and methods)
3. Models
 - a. Overview
 - b. Authentication (login stuff)
 - c. Authorization (access control stuff)
 - d. Workorders
 - e. Workflows
 - f. Inventory
 - g. Orders
 - h. Project Management
 - i. Notifications (TBD)
 - j. Reporting (TBD)

Notes

- System architecture:
 - o Overall components (diagram)
 - Frontend, backend, database, integrations
 - o API Interface
 - Resources and methods
- Tooling:
 - o Frontend:
 - Vue, Webpack, Babel, Axios
 - SASS, Bootstrap
 - Testing:
 - Jest with Vue Test Utils (and maybe Sinon.js for faking, TBD)
 - o Backend:
 - ASP.NET Core 2.2
 - EF Core 2.2
 - Testing:
 - xUnit for unit testing
 - TestServer (built in I think) for automated integration/API testing
 - Postman for manual integration/API testing
 - o Database:

- PostgreSQL? Will determine
- Continuous Integration:
 - Still TBD
 - Jenkins with Docker most likely
- Deployment:
 - TBD
 - Likely to have server deliver Webpack package of frontend for client rendering
 - Will require installation executable of server with configuration wizard
- Models:
 - Full view (maybe a more generic view without all the fields, just the model names, to make it fit)
 - Authentication/Authorization (In progress)
 - Accounts & Role-Based Access Control (In progress)
 - Workorders
 - Workflow
 - Inventory
 - Orders
 - Project Management
 - Notifications (TBD)
 - Reports (TBD)

Test Plan & Execution

February 18, 2019

12:29 PM

Business Plan

March 14, 2019 16:15

Notes

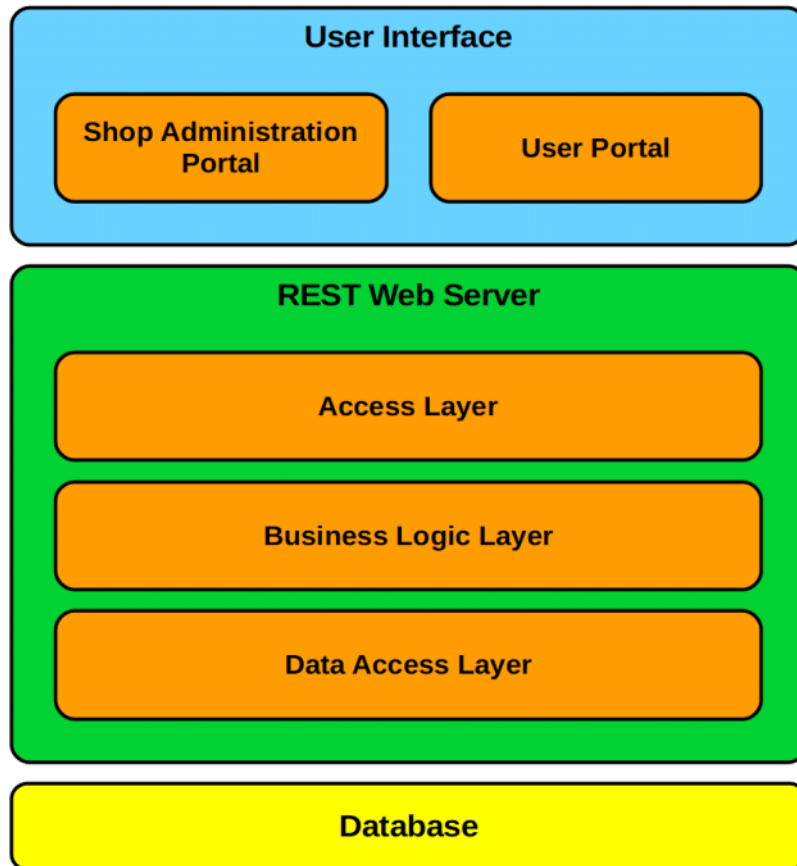
- Why this is going to be a thing
- Describe current system to define general purpose
- List all components and what they correlate to in the current system
 - o Describe the use from the current system and how the new system applies to that original use case
- Describe current system data/work flows and how the flows change in the new system
 - o Describe the economical impact of the updated flow/system
- How much this software is worth to the workshop
- Future profitability to future clients
 - o Try to use concrete(-ish) numbers to describe time savings of paper-based systems vs this new web system

Examples

- Discuss time tracking - how only billable hours were really tracked, maybe only accounted for 1/3 of a day - where does the other two-thirds of the day go?
- Discuss time spent managing managerial tasks, managing paper documents, looking old documents up
- Discuss time spent looking up materials
 - o Materials may have been used in previous workorders, but searching for that workorder to find the source of the material is slow
 - Inventory system allows for the tracking of all of this information

Overview

October 24, 2018 9:16 AM



Frontend

- Vue interface
 - o Server-side rendering

Backend

- REST web server
 - o ASP.NET C#
 - o Entity Framework
- SQL Database

Server

Wednesday, November 21, 2018 9:25 AM

- ERP.Common
 - Common files and interfaces that may need to be used in a number of projects
 - May not be necessary for our purposes
- ERP.Models
 - Objects that define our models/entities
- ERP.Services
 - Business logic after routing
 - This would likely be best suited if we converted to DTOs instead of always using the models in the API layer
 - The following link gives an idea of what the service layer is in respect to the API/Controllers layer:
 - <https://stackoverflow.com/questions/31185072/effectively-use-async-await-with-asp-net-web-api>
- ERP.Repositories
 - Data access layer, handles all database interfacing
- ERP.API
 - Endpoint controllers
 - Startup project

Authentication & Authorization

February 18, 2019 3:41 PM

Requirements

- Role-based access control with permissions and roles
- Accounts
 - o Primary "super-user" or "administrator"
 - o Ability to create accounts given to administrator, can't just register
 - o Unknown requirements given workorder submission

OAuth

- Grant type = password grant
 - o Client credentials that are stored with the server can be trusted within the first party client

Tooling

- ASP.NET Identity for authentication and account management
 - o Uses OAuth2
 - o Utilizes OWIN
 - Will need to leverage OWIN for token based authorization

Notes

- Authentication = logging in, validating an account
- Authorization = determining if a request is valid and is allowed to access the requested resource
- OAuth2 = authorization and access control
- OpenID = authentication on top of OAuth2

Links

- ASP.NET Identity:
 - o <https://docs.microsoft.com/en-us/aspnet/identity/overview/getting-started/introduction-to-aspnet-identity>
- OWIN & OAuth2.0 Authentication:
 - o <https://docs.microsoft.com/en-us/aspnet/aspnet/overview/owin-and-katana/owin-oauth-20-authorization-server>
- OpenID and IdentityServer4:
 - o <https://christianlydemann.com/creating-an-openid-connect-system-with-angular-5-and-identityserver4-oidc-part-1/>

Notes from meeting

- SAML
 - o Library authentication

Links

February 19, 2019 11:27 AM

Security

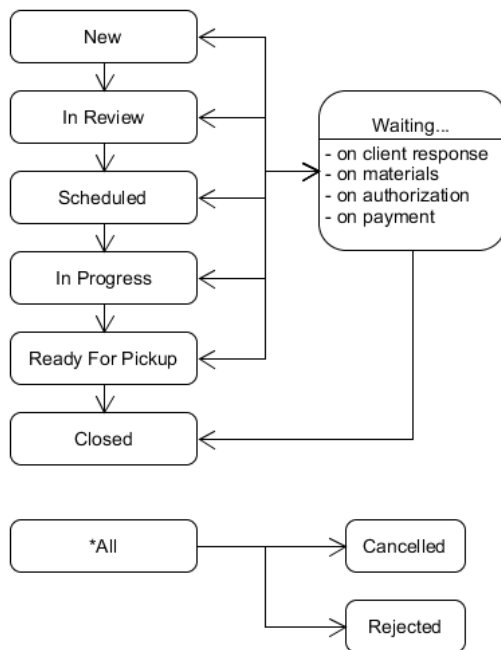
- https://github.com/OWASP/CheatSheetSeries/blob/master/cheatsheets/REST_Security_Cheat_Sheet.md

For Future Consideration

- Argument for Inversion of Control (via CastleWindsor):
 - o <https://stackoverflow.com/questions/124871/what-is-castle-windsor-and-why-should-i-care>

Workorder Flow

October 31, 2018 11:08 AM

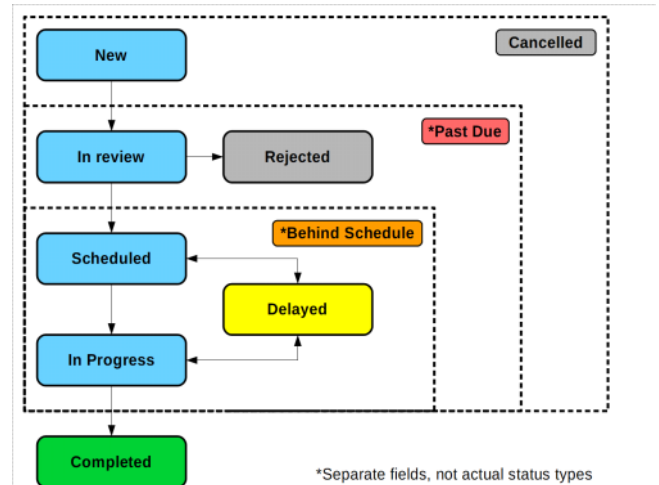


Questions

- What should the status flow of a workorder look like?
 - o Workorder needs approval before work can begin - does it get approved before you review it, or after? Before or after estimates/quotes are made?

Notes

- Maybe change so that "Rejected" can only come from "In Review" like original

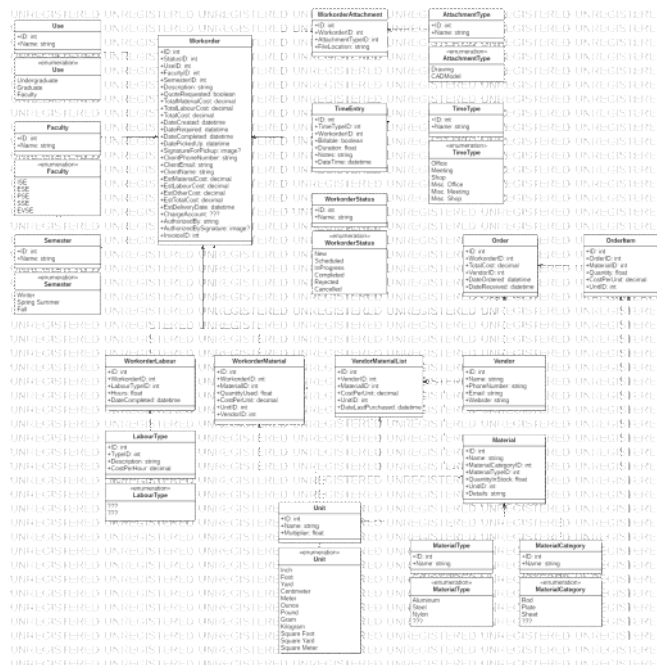


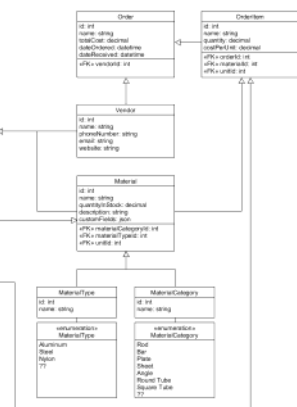
Questions

October 24, 2018 9:17 AM

Questions

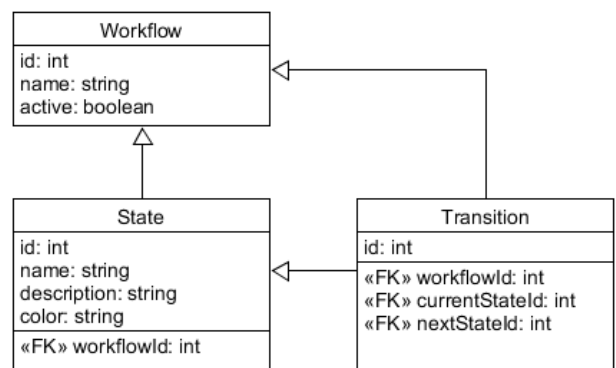
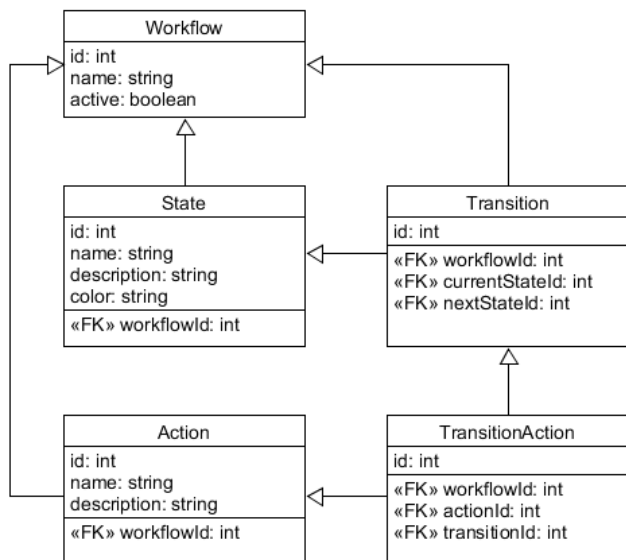
- Does the university have an OAuth server?
 - Will we need to leverage this for the intranet site?
 - Should we use it to allow shop administrators to sign into the shop interface?
 - If we did this, we'd need a global admin account created by the university that can manage what other accounts have access. Will need to follow up with it.
 - Ask Chris how he'd prefer to log in





Workflows

Thursday, February 7, 2019 6:47 PM



Workorders

October 31, 2018 8:42 AM

Workorder

- ID (guid)
- Status
- Use (undergrad, grad/research, other)
- Faculty
- Semester
- Sequential ID (in semester?)
- Description
- Quote requested
- Total material cost
- Total labour cost
- Total cost
- Date requested
- Date required
- Date completed
- Date picked up
- Signature for pickup (?)
- Client phone #
- Client email
- Client name
- Estimated material cost
- Estimated labour cost
- Estimated other costs
- Estimated total cost
- Estimated delivery date
- Charge account
- Authorized By
- Authorized By Signature
- Invoice ID (separate Invoice table?)

Labour Type

- ID
- Type
- Description
- Cost per hour

Workorder Status

- ID
- Name

ID	Name
1	New
2	Scheduled
3	In Progress
4	Delayed
5	Behind Schedule
6	Completed
7	Rejected
8	Cancelled

Workorder Attachment

- ID
- Workorder ID
- Attachment type
- File Location

Attachment Type

- ID
- Name

ID	Name
1	Drawing
2	CAD File

Workorder Material

- ID
- Workorder ID
- Material ID
- Quantity used
- Cost per unit

Workorder Labour

- ID
- Workorder ID
- Labour Type ID
- Hours
- Date completed

Notes

- Labour cost per hour depends on if it is a student or not as well as how many hours that person has already utilized
 - o Students get 10 hours at \$5/hour
 - o Standard is \$60/hour
 - o *Price override? Additional labour type for discounted?*
 - o *Will require keeping track of clients and number of hours already utilized*
 - o Does this only count for direct shop time?
- Should save material/labour cost per workorder in case those cost values change in the future
- Change "Delayed" and "Behind Schedule" to separate flags, not statuses
 - o Add "On Schedule" to this

Questions

- Will there be more than one type of labour or is the time type "shop" good enough?
 - o Eg. Drilling, milling,
- What type of file types should be expected?
- Any purpose in storing semester or sequential id within it? Probably, yes.
 - o Semester could either be manually put in as a function of the app *OR* it could be a computed column based on date range per semester
 - o Should "Semester" be "Semester ID" with an associated "Semester" column (eg. 1 == Winter, 2 == SS, 3 == Fall)
 - o Sequential ID could simply be replaced by sorting by date
- For "Status", should "Behind Schedule" be a separate field to allow any status to still be "Behind Schedule"
 - o Eg. "In Progress" and "Behind Schedule"
 - o Should "Behind Schedule" be for Chris' schedule only, and another status (eg. "Past Due") be for the requested due date?
 - "Past Due" and "Behind Schedule" can both be calculated values based on due date and scheduled date (?), respectively
 - What should schedule dates look like?
 - ◆ Set start date with an estimated time-to-completion?
- Should we be adding clients to a table (with their contact info and their hours utilized)? Is this worth collecting?
 - o I'd imagine *most* clients are 1-3 times (for capstone)
- Do we need an invoice table?
 - o What information all goes into the invoice?
 - o Does Financial Services give an invoice number, or do you make that?
- Clients
 - o Will client information be an object received from intranet? What's the story going to be with this?

October 31, 2018 8:42 AM



- ## Notes

- ### Material

- Material Type**

- Material Category**

- | ID | Name |
|----|-------|
| 0 | Rod |
| 1 | Plate |

Vendor

- ### Material Vendor List

- ID (int)
- Material ID
- Vendor ID
- Cost per unit
- Date last purchased

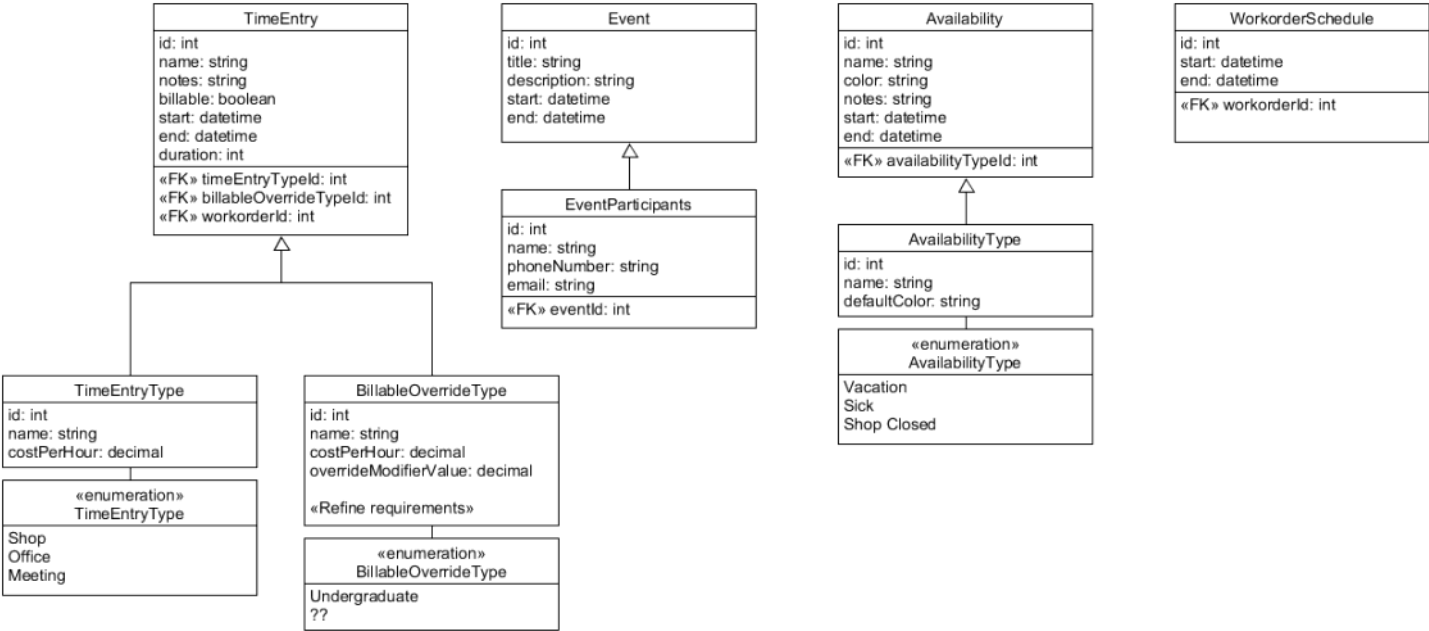
Order

- Order Item**

- ID (guid)
- Order ID
- Material ID
- Quantity
- Cost

Project Management

October 31, 2018 8:42 AM



Time Tracking

October 31, 2018 8:46 AM

Time Entry

- ID
- Time Type ID
- Workorder ID
- Billable
- Duration
- Clients/Parties/People
- Notes
- Date/Time

Time Type

- ID
- Name

ID	Name
0	Office
1	Meeting
2	Shop
3	Out of Office
4	Sick
5	Vacation
6	Lunch
7	Break

Questions

- Will the misc. time entries always be non-billable?
 - o Should billability be based on time-type or based on each time entry?
 - More flexible if based on time entry, but if misc. is always non-billable then it could just be tied to time-type
- Should we break down the time types even more?
 - o Shop maintenance
- How would Chris like to enter time?
 - o Start/end time?
 - o Number in hours?
- Should there be a way to tie this together with **Labour Type** in order to better track labour on each project?
 - o It seems like it probably should be

Time entries and labour

- Is all labour billed at the same rate?

Notes

- "Client" depends on time type
 - o Meeting needs it
 - o Shop just relies on workorder or misc.
 - o Office does not need it

Idea

- Add time entry
 - o From workorder - automatically associates it with the workorder
 - o Elsewhere - section to add associated workorder
- If "shop" type, set labour type
 - o Add labour types for maintenance and such
 - o Any need for misc.?
 - Basically, if it's not associated to workorder, it is misc.
 - What about office?

Reports

October 31, 2018

2:32 PM

General Website

October 31, 2018

8:42 AM

- Accounts

Notes

October 31, 2018 8:46 AM

- Arguments for so many reference tables for relatively static data
 - Provides extensibility and customizability should that ever be needed
 - Useful for many of our tables such as the categories and types tables
 - Not so useful for the semesters table for example
 - Is an actual pattern, libraries can handle it and other programmers will recognize it
 - Performance hit is nearly imperceptible as these small tables that are used constantly are cached

Links

February 17, 2019 10:08 PM

API Resources

- <https://opensource.zalando.com/restful-api-guidelines>
- **HATEOAS: (important: read and implement in backend)**
 - o <https://restfulapi.net/hateoas/>
 - o <https://blog.jeremylikness.com/5-rest-api-designs-in-dot-net-core-2-ad2f204c2d11>
- **RBAC:**
 - o <https://cloudify.co/2016/04/15/simple-secure-role-based-access-control-rest-api-rbac-server-devops-cloud-orchestration.html>

Notes

March 14, 2019 11:33 AM

- Considering *validation logic*: perform checks to see if everything exists and add to `ModelState.Errors` (or equivalent), then return `NotFound` if not empty (or `BadRequest`?)
- HTTP Responses:
 - Currently returning strict JSON
 - Future:
 - HATEOAS with HAL (`application/hal+json`)
- HAL:
 - `_links`:
 - Self/possible actions (HATEOAS)
 - `_embedded`:
 - Actual data/resource/collection
 - Fields:
 - Miscellaneous fields regarding the state of the requested resource
 - Eg. Number of items included in `_embedded` collection, total number of items in resource database, etc.

Accounts & Role-Based Access Controls

February 17, 2019 11:13 PM

Accounts

-

Workorders

Thursday, February 7, 2019 5:35 PM

Workorder Actions

- Workorder & details:
 - o `/api/workorders`
 - GET - get all workorders
 - POST - create new workorder
 - o `/api/workorders/{workorder-id}`
 - GET - get {workorder-id} and all arrays
 - PATCH - update simple details for {workorder-id}, not any of the linked arrays
 - DELETE
- Estimate:
 - o `/api/workorders/{workorder-id}/estimates`
 - GET - get all estimates for {workorder-id}
 - POST - create new estimate for {workorder-id}
 - o `/api/workorders/{workorder-id}/estimates/{estimate-id}`
 - GET
 - PUT - update estimate {estimate-id}
 - DELETE
- Comment:
 - o `/api/workorders/{workorder-id}/comments`
 - GET
 - POST
 - o `/api/workorders/{workorder-id}/comments/{comment-id}`
 - GET
 - PUT - replace {comment-id} on edit
 - DELETE
- Note:
 - o `/api/workorders/{workorder-id}/notes`
 - GET
 - POST
 - o `/api/workorders/{workorder-id}/notes/{note-id}`
 - GET
 - PUT
 - DELETE
- Material:
 - o `/api/workorders/{workorder-id}/materials`
 - GET
 - POST
 - o `/api/workorders/{workorder-id}/materials/{material-id}`
 - GET
 - PUT
 - DELETE
- Attachments:
 - o `/api/workorders/{workorder-id}/attachments`
 - GET
 - POST
 - o `/api/workorders/{workorder-id}/attachments/{attachment-id}`
 - GET
 - PUT - replace attachment information (will also require file upload, then server will update location)
 - DELETE
- Time entries: *(this is likely to change, see notes)*
 - o `/api/workorders/{workorder-id}/time-entries`
 - GET
 - POST
 - o `/api/workorders/{workorder-id}/time-entries/{time-entry-id}`
 - GET
 - PUT
 - DELETE
- State and transitions:
 - o `/api/workorders/{workorder-id}/state`
 - GET - retrieve full state definition
 - POST - update the workorder state with a valid state and add to the transition history
 - o `/api/workorders/{workorder-id}/state/history`
 - GET - retrieve state/transition history
 - o `/api/workorders/{workorder-id}/state/transitions`
 - GET - retrieve possible state transitions

Search and Filtering

- Search:
 - o `/api/workorders/search?{params}`
 - GET - takes query string that will search title, description, id, client
- Filter:
 - o `/api/workorders/filter?{params}`
 - GET - takes query string that will filter by state, faculty, semester, etc.

Rationale

- PATCH if we need to update an array within an endpoint, PUT if we have an endpoint directly to the item within the array

Notes for future thoughts

- Using PATCH for any of these tables means if multiple of the same request get sent, it'll update many times
 - o For example, consider Estimates: An estimate will contain a list of materials within it, so a PATCH request will be used to add to it. If the request gets sent numerous times, many of the same material will get added, giving an incorrect estimate.
 - A bonus to PATCH is we can update the resource as soon as a single material is added, meaning progress isn't lost when entering numerous
 - With PUT, you'd have to "Edit" the resource, add the materials, then "Accept" the changes before a PUT is made
- For time entries:
 - o Is this one best practice, or should it just use the "`/api/time-entries`" endpoint?
 - To update a certain time entry for a certain workorder, you'd:
 - GET `/api/time-entries?workorderId={workorder-id}`
 - PUT `/api/time-entries/{time-entry-id}`

Inventory

February 12, 2019 2:51 PM

Inventory

- Materials:
 - /api/inventory/materials:
 - GET
 - POST
 - /api/inventory/materials/{material-id}:
 - GET
 - PUT
 - DELETE
- Material Types:
 - /api/inventory/materials/types:
 - GET
 - POST
 - /api/inventory/materials/types/{type-id}:
 - GET
 - PUT
 - DELETE
- Material Categories:
 - /api/inventory/materials/categories:
 - GET
 - POST
 - /api/inventory/materials/categories/{category-id}:
 - GET
 - PUT
 - DELETE
- Vendors:
 - /api/inventory/vendors:
 - GET
 - POST
 - /api/inventory/vendors/{vendor-id}:
 - GET
 - PUT
 - DELETE
- Orders:
 - /api/orders
 - GET
 - POST
 - /api/orders/{order-id}
 - GET
 - PUT
 - DELETE
- Order Items:
 - /api/orders/{order-id}/items
 - GET
 - POST
 - /api/orders/{order-id}/items/{item-id}
 - GET
 - PUT
 - DELETE

Project Management

February 12, 2019 2:51 PM

Project Management

- Calendar:
 - /api/calendar
 - GET - retrieve all calendar objects (time entries, events, availability, workorder scheduling)
- Time Entries:
 - /api/calendar/time-entries
 - GET
 - POST
 - /api/calendar/time-entries/{time-entry-id}
 - GET
 - PUT
 - DELETE
- Events:
 - /api/calendar/events
 - GET
 - POST
 - /api/calendar/events/{event-id}
 - GET
 - PUT
 - DELETE
- Availability:
 - /api/calendar/availability-entries
 - GET
 - POST
 - /api/calendar/availability-entries/{availability-entry-id}
 - GET
 - PUT
 - DELETE
- Workorder Scheduling:
 - /api/calendar/workorders
 - GET
 - POST
 - /api/calendar/workorders/{workorder-id}
 - GET
 - PUT
 - DELETE

Standards

Tuesday, December 4, 2018 3:25 PM

UI Standards

- Fonts:
 - o Heading: Montserrat
 - o Standout (i.e. navigation): Montserrat
 - o Content: Roboto
- Color Palette: **(Note: this requires much fine tuning, but the top blue is the current main scheme. Need to fine tune and determine auxiliary colors)**

8796A4	607589	425872	2C4358	19344B
FEF2CD	D3C38F	B09D60	887740	746021
FEDCCD	D3A48F	B07960	885640	743B21

o

FED0CD	D3948F	B06660	88453F	742721
FEE4CD	D3AF8F	B08660	88623F	744921
7E989B	587C81	3C666C	284E53	154147
9DC2A3	6DA176	498754	30683A	195924

Views

October 31, 2018 9:15 AM

Overview

- Dashboard/Home
- Project Management
 - o Calendar
 - Daily/Weekly
 - Time Entries
- Workorders
 - o Workorder List
 - o Workorder Queues
- Materials
 - o Materials List (all available materials)
 - o Materials Inventory (current inventory)
 - o Orders (?) or Invoices
 - o Vendors
- Reports
 - o Reports Dashboard
 - o Reports List
- Settings

Project Management Calendar

- View scheduled workorders
- View time entries for any given day
- Enter new time entries

Resources & Endpoints

Thursday, November 8, 2018 11:16 AM

- Workorders
- Notices of projects
- Reports
- Materials
- Vendors
- Time entries (?)

Workorders

- /workorders
 - o GET - get all workorders
 - o POST - create new workorder
- /workorders/{id}
 - o GET - get workorder {id}
 - o PUT - update workorder {id} if using "edit" mode
 - o PATCH - update specific fields of workorder {id}
 - Status updates
 - o DELETE - delete workorder {id}
- /workorders/{id}/status
 - o PUT - update status of workorder {id}
- /workorders/{id}/materials

Materials

- /materials
 - o GET - get all materials
 - o POST - create new material
- /materials/{id}
 - o GET - get material {id}
 - o PUT - update material {id}
 - o DELETE - delete material {id}
- /materials/type/{type_id}
 - o GET - materials of type {id}

Vendors

- /vendors
 - o GET - get all vendors
 - o POST - create new vendor
- /vendors/{id}
 - o GET - get vendor {id}
 - o PUT - update vendor {id}
 - o DELETE - delete vendor {id}

Project Management/Calendar/Time Tracking

- /calendar
- /calendar/events
- /calendar/availability
- /calendar/workorders
- /time-entries
 - o GET - get all time entries
 - Date - get time entries from a certain MM-DD-YYYY
 - Mindate & maxdate - get date range from mindate to maxdate

Notes

- Use JSON Merge Patch or idempotent PUT @ /status to update status?
 - o How about materials? Labour and time?
- Think about filtering/sorting/pagination
 - o Eg. To request all materials from vendor MatsNStuff, ID = 12:
 - /materials?vendor_id=12
 - /vendor/12/materials

Notes

- Four aspects of the calendar:
 - o Events
 - o Availability
 - o Time tracking/entries
 - o Workorder/capacity planning
- How should we store and deliver the calendar information?
 - o /calendar/2019/jan
 - /calendar/2019/jan/events
 - /calendar/2019/jan/availability

General Application Design

November 8, 2018

3:53 PM

Possible Options

- Navigation bar
 - o Position
 - Side
 - Top
 - o Type
 - Icons (shrinkable/expandable)
 - Full text
- Additional Features
 - o On-demand project management side bar
 - o Adjustable pages (rearrange cards, etc.)

Notifications

- Toast notifications:
 - o Adding/deleting/updating:
 - Time entries
 - Workorders
 - Materials
 - Etc.
- Notification tray:
 - o New incoming workorders
 - o General calendar notifications
 - o New automatic report generated
 - o Reminders?
 - Eg. Self-set reminder on a workorder or material order to take another look/update details

Styling

Monday, January 21, 2019 1:59 PM

Jan 21st

- Not sold on shadows (see current side menus)
 - Perhaps flat would be easier to make consistent while providing a simpler and 'easier on the eyes' design
 - Consult with others
- Not sold on position of expand/retract button on side menus
 - Overlaps with the two links near it when they are active; does this look bad?
 - Also forces the active/hover background to be 20px shorter than the full width of the expanded menu; again, does this look bad, or like a stylized feature?
 - Consult with others

University of Regina

Home	Workorders	Inventory	Projects	Schedule	
-------------	-------------------	------------------	-----------------	-----------------	--

Home

- Workorders

- Inventory

- Projects

November 2018

S	M	T	W	T	F	S
6	5	4	3	2	1	
13	12	11	10	9	8	7
20	19	18	17	16	15	14
27	26	25	24	23	22	21

Recent Submittals

Date	Status
02-Nov	Open
17-Sep	Open
03-Oct	Open
01-Oct	Open

[illegible]

The sketches illustrate a software interface for a 'Library' application. The main window features a menu bar with 'Library', 'Window', and 'Help' options. Below the menu is a toolbar with icons for file operations (New, Open, Save, Print, etc.). The main area is a large text field. A secondary window, titled 'Books', displays a table of book records with columns for ID, Title, Author, and Date. A third window, titled 'Details', shows the information for a selected book, including its title, author, and date.

Date	Description	Amount
10/10/10	100 shares	100.00
10/11/10	100 shares	100.00
10/12/10	100 shares	100.00
10/13/10	100 shares	100.00
10/14/10	100 shares	100.00
10/15/10	100 shares	100.00
10/16/10	100 shares	100.00
10/17/10	100 shares	100.00
10/18/10	100 shares	100.00
10/19/10	100 shares	100.00
10/20/10	100 shares	100.00
10/21/10	100 shares	100.00
10/22/10	100 shares	100.00
10/23/10	100 shares	100.00
10/24/10	100 shares	100.00
10/25/10	100 shares	100.00
10/26/10	100 shares	100.00
10/27/10	100 shares	100.00
10/28/10	100 shares	100.00
10/29/10	100 shares	100.00
10/30/10	100 shares	100.00
10/31/10	100 shares	100.00
	Total	2000.00

Student Side

University of Regina

New Workorder
2018-51-001

Project Scope

Class/Lab

Phone #

Job Description

Date








Requisitioned By

Email

Drawing Supplied? ☐ DWG#

Date Required

Digital Signature

Chris' Sign	
	New Wood
	Estimated
	Estimated
	Estimated
	Estimated
	Estimated
	TOTAL

de

University of Regina	
Prkorder Request	# 2018-S1-001
Cost of Labour	Date
Cost of Material	
Other Costs	
Cost of Labour	
ESTIMATED COST	Status
	Approved for further Meeting
	Send

Workorders

October 31, 2018 9:56 AM

Requirements

- Sorting
 - Arrange by column header
 - Tabs to view by selected status? Cards for each status?
- Searching
 - Search bar
 - Advanced search w/ options

Project Management

December 30, 2018 4:40 PM

Time Entry

- Dropdown for current/active workorders
- Time entry options:
 - o Start/stop time
 - o Start time and duration
- Dropdown for time type
- Dropdown for billing rate modifier (if time type = shop time)

Calendar

- <https://nick-basile.com/blog/post/building-a-calendar-with-vuejs>

PM Quick Access Panel

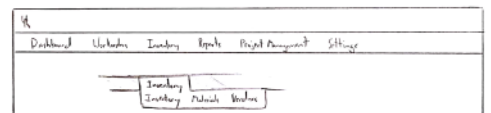
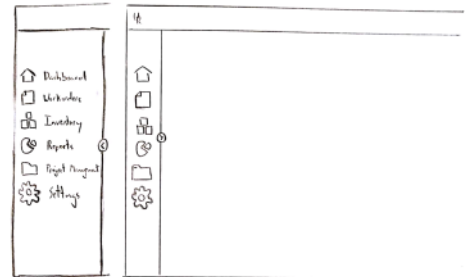
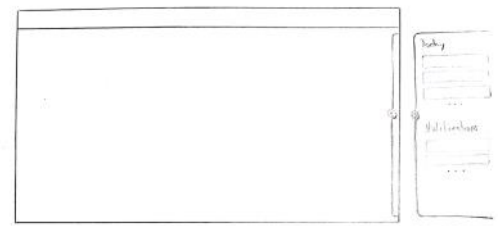
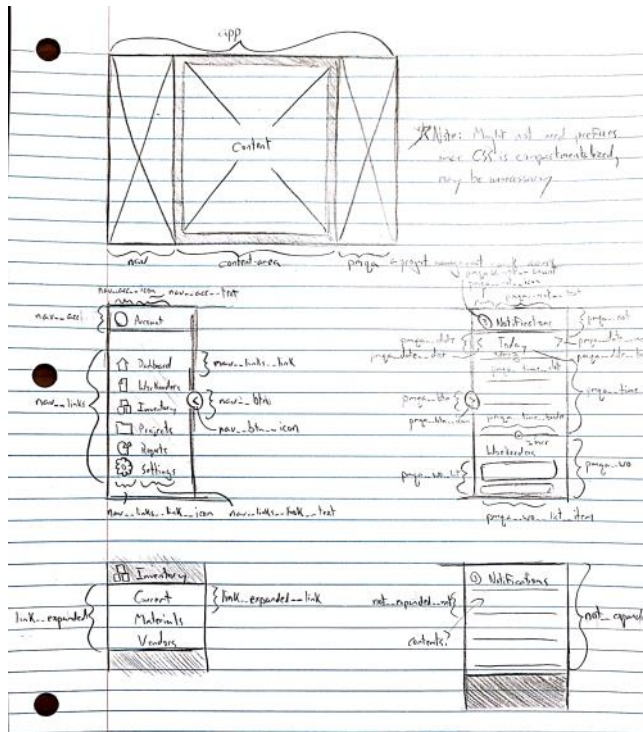
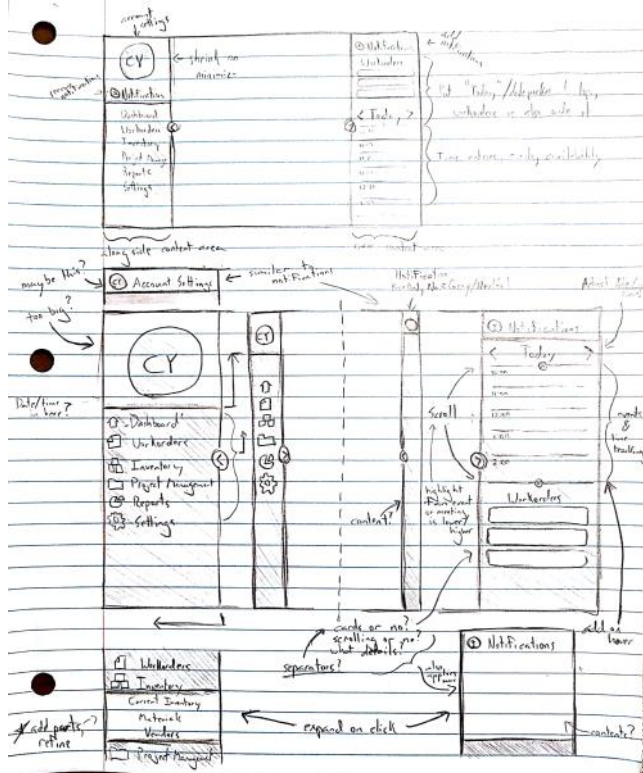
- Collapses to nearly completely retracted state (maybe 14-18px?)
 - o If there are notifications, have a block at the top extend out with a red circle and the number of notifications
 - Clicking this extends the PMQA panel and immediately expands the "Notifications" section
 - o If no

Navigation

Thursday, January 17, 2019 8:46 PM

Calendar						
November 2018						
S	M	T	W	T	F	S
					1	2
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	

Recent Submissions	
Order	Status
FS-01-18	✓
FS-02-18	✓
FS-03-18	✓
SS-03-18	✓



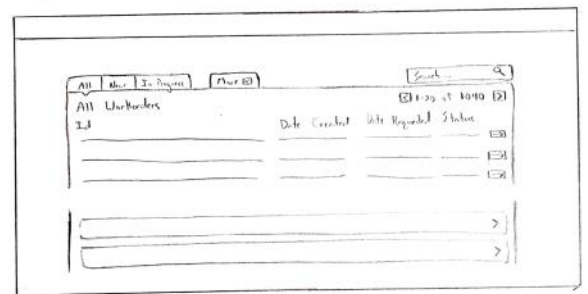
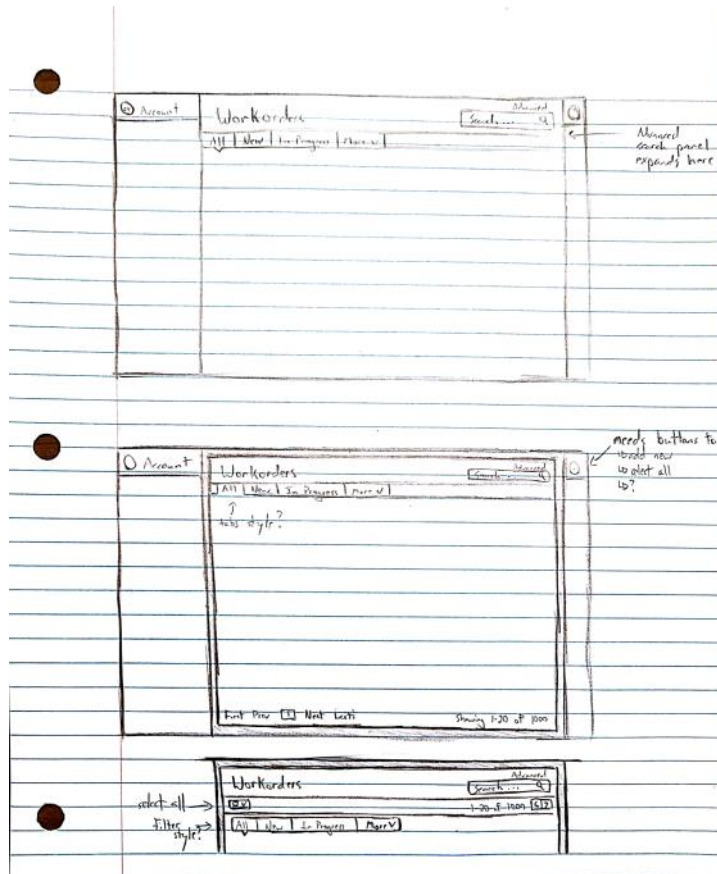
Inventory

January 18, 2019

8:54 AM

Workorders

Thursday, January 17, 2019 8:52 PM



Thursday, January 17, 2019 8:52 PM

Thursday, January 17, 2019 8:52 PM

Monthly view:

- ↳ events
- ↳ cap. planning
- ↳ availability

Notes:

- ↳ should there be separate bus tracking vs. cost calculation?
- ↳ the costs, but time entry, but other things to be still difficult

for other events

color coded busier specified

Workorder placing

All-day events

Time entries, daily breakdown

Calendar/workorder

Task entries vs. events vs. capacity planning vs. availability

vt/calendar/daily/8 day?

↳ reduces time entries, events, workorder, availability

Settings Requirements

February 20, 2019

12:56 PM

Categories

- Accounts:
 - Users
 - Roles & Permissions
- Workorders:
 - Workflows
 - Miscellaneous:
 - Faculties
 - Uses
- Inventory:
 - Material Types
 - Material Categories
 - Units
- Billing:
 - Billing Values and Overrides

Testing Plan

Friday, December 7, 2018 2:10 PM

Test Plan

- Frontend:
 - o Jest with Vue Test Utils
 - Sinon.js for fakes? TBD
 - o Potentially Selenium or Cypress for automated UI testing (time permitted)
- Backend:
 - o xUnit for unit testing
 - o Postman for manual integration/API testing
 - o Microsoft's TestServer for automated integration/API testing
 - o <https://code-maze.com/unit-testing-aspnetcore-web-api/>
 - o <https://tech.trailmax.info/2013/04/autofixture-and-moq-to-test-entity-framework-project/>
- Continuous Integration:
 - o Undetermined needs
 - o Jenkins with Docker (?)
 - <https://jenkins.io/doc/tutorials/build-a-node-js-and-react-app-with-npm/>

Benefits of CI/CD with Jenkins

- Create pipeline to ensure the automated tests are run on every commit
- Can deliver application for client/presentation/advisor purposes

Up Next

Sunday, March 31, 2019 9:51 AM

- Rework controllers to utilize services instead of repository
 - o Rationale:
 - Separation of concerns - API shouldn't directly be manipulating database, it is simply the interface with which external applications communicate with the server
 - Maintainability - allows for cleaner and more succinct code and function names, and it makes the repository functions more reusable, meaning better DRY-ness
 - Extensibility - easier to add new controllers and new endpoints
 - Validation - significantly easier to implement custom validator system
 - o Additional - only utilize DTOs in the controller, no actual models:
 - Further separates DB from API
 - Allows for more specific and appropriate information for each API, only requiring/returning what is needed for that endpoint
- Rework endpoint/repository validation
 - o Create a validator object that takes
- Implement frontend caching using something like LocalForage
 - o Rationale:
 - Increases performance and decreases requests needed by caching commonly needed data
 - Would primarily be used for filters, states, categories, etc. - could also cache most recent or most common materials

Public

Sunday, March 31, 2019 1:55 PM

Discussion Points

- Development process
 - o Design and design decisions
 - Show low fi, high fi
- Tools, frameworks, languages, etc.
- Client requirements and features
 - o Translation of requirements to required features
- Show applications, demo
- Discuss what features were implemented and what requirements were fulfilled
- Future work, next steps and sprints (high-level overview)
- Business plan/sense
- Pitfalls
 - o Scope creep

Development Process

- Agile
 - o Frequent meetings with client
 - o Break features up into time slots aka sprints
- Gather requirements
 - o Form features from requirements
- Choose technologies and frameworks
- Design models
- Design API
- Design UI
 - o Lo-fi iterations, actual

Technical

Sunday, March 31, 2019 1:55 PM

Discussion Points

- Purpose and Scope
- Project Management
 - o Client requirements, overall features
 - o Sprints
- Development strategy and where it deteriorated
- Tools, frameworks, and languages
- System Design
 - o Overall Architecture
 - o UML
 - o Frontend, Vue, etc.
 - o Backend project architecture, data flow, ASP.NET Core
 - o API design
- Testing
 - o Manual API testing
 - o Example unit testing, mocking, plan for proper implementation
- State of project:
 - o What is implemented
 - o Future work
- Future work
 - o Design that we didn't get to in time
 - o Deployment plans

Development Strategy

- Agile
 - o Originally divided up the work, wasn't strict
 - o Planned sprints, met periodically with client
 - o Deterioration:
 - Division of labour became muddled
 - Originally, K was backend, J was frontend
 - Both became more full stack
 - J handled model/API design
 - K handled official documentation
 - Sprints were falling behind
 - Seemingly straightforward features had hidden depth to them
 - New features were realized along the way, causing redesigns of some components - not everything is perfect even still
 - Was good at checking design with client, periodic meetings helped him to more fully realize what features he wanted/needed

Tools, Frameworks, Languages

- Describe what the technologies are and why we chose them
- Backend:
 - o ASP.NET Core, Entity Framework Core, SQL LocalDB
 - o Notes:
 - Intend on using a proper SQL, likely PostgreSQL
 - Reasoning: only made development more difficult since it is still changing organically, will build out properly for deployable
- Frontend:
 - o Vue, Vue Router, Bootstrap-Vue, Webpack
 - o Axios, DayJS (maybe not really necessary besides a cursory glance)
 - o Sass
 - Show image of Sass
- Development Tools:
 - o Postman
 - Testing, database populating
 - o ZenHub
 - Originally started using it for managing epics, tracking issues, etc.
 - Filled it too early, didn't use it much
 - Began using GitHub issues and projects recently
- Testing:
 - o Talk about this here, or in future works?
 - o Backend: xUnit, AutoFixture, FakeItEasy
 - o Frontend: Jest
 - o Integration: Semi-automated Postman collections

System Design

- Overall Architecture:
 - o How the separate parts communicate
 - o Integrations
- Components:
 - o Discuss the breakdown of individual components
- UML:
 - o Discuss the model diagram, show the overall and the individual ones
- Backend:
 - o Project structure, purpose of each project
 - o Components
 - o Dependency Injection
- Frontend:

Time Notes-Total 20 mins

- Slide 1 Introduction - 1 minute
- Slide 2 Purpose and Scope - 1 minute
- Slide 3 Requirements - 1 minute
- Slide 4 Development Strategy - 2 minutes
- Slide 5 Sprints - 1 minute
- Documentation and Log Book
- Slide 6 Development Difficulties and Challenges - 2 minutes
- Slide 7 System Design (intermediate)
- Slide 8 Overall Architecture - 1 minute
- Slide 9 Frontend and reasons why - 2 minute
- Slide 10 Backend and reasons why - 2 minute
- Slide 11 Database Design - 2 minute
- Slide 12 Testing (intermediate)
- Slide 13 API Testing - 2 minutes
- Slide 14 API Tests in Action (intermediate)
- Slide 15 Testing in the Future - 1 minute
- Slide 16 Future Plans - 2 minutes
- Slide 17 Questions?
- Slide 18 Appendixes
- Slide 19 Thanks to ...

Why?

- Backend:
 - o ASP.NET for the maturity of tooling around it
 - o C# for ASP.NET and for Linq
 - o EF Core for Code First
 - o LocalDB for ease of development, deleting and recreating database as models change
- Frontend:

- Basics about structure of site
 - How we get and use data
- API:
 - General endpoints, briefly discuss HTTP methods

Testing

- Manual Postman requests:
 - Manually verify data is as expected
 - Allowed us to see what was being returned
- Arrange, act, assert:
 - Autofixture with Moq to take care of the arrange

Future Work

- Proper separation of API controllers and repository layer using the service layer
- Creation of tailored DTOs to tailor what information is needed from and given to each endpoint
- Scalable and expandable validation system:
 - Strategy/composite design pattern, ValidatorFactory, dependency injection
- Integrations with U of R's SAML authentication server and Financial Services
- Dashboard, report generation, account management, user submission (comes with login capabilities)
- Settings, full customizability, workflow engine
- Installable deployable