Sprint 1 Retrospective

19/09/2020

Members Present: Jerrayl, Mehmet, Juwey, Ajay, Winnie

Goals: Implement a prototype login system (to be replaced in next sprint), set up architecture and dependencies, code CRUD operations for portfolios, including retrieving individual portfolios, all portfolios created by a particular user, and individual pages from a portfolio.

| Requirement	Result
● Landing Page	Complete → A landing page with a call to action is displayed when the user enters the website, allowing users to register to start creating their ePortfolios easily.
● Prototype login system	Complete → Currently a placeholder for account management. On logging in, the user will be directed to their dashboard. Will be using Google OAuth in the second sprint.
● Architecture and dependencies (frontend)	Complete → Set up the frontend to use redux state management so state is managed on the app level and not a component level. Use MaterialUI for display components and styles. Set up backend using express and connected it to MongoDB database.
● Blog system	Not Finished → It was not part of the MVP, not a priority. Part of blog functionality was later subsumed under the grid system.
● Create portfolios	Complete → On the dashboard, on clicking the plus card, users will be directed to type in the desired name of the ePortfolio, go to the next page, select a template to use, or blank to start from scratch.
● Edit portfolio	Semi-complete → The display is finished, but is not fully functional. The functionality will be the highest priority to implement in the second sprint.
● Display portfolio from link	Complete → Guests can view an ePortfolio with a link from the owner of the ePortfolio.

	Users can get that link from a drop down menu on each ePortfolio on the dashboard. On clicking 'Get Link', a modal will pop up with a button to copy the link to the clipboard.
● Delete an ePortfolio	Complete → On each ePortfolio on the dashboard, users can open a drop down menu and click delete to delete that specific ePortfolio.
● Display portfolios on dashboard	Complete → Existing ePortfolios of a user is displayed from the database on their own dashboard.
● Comment system	Semi-complete → The comment system is implemented such that users can make comments in a comment box. However, it has yet to be integrated into the web application, and will be the top priority in sprint two, together with the edit portfolio functionality.

Technologies Introduced

Client-side Dependency	Purpose
material-ui	React component library
react	User interface JavaScript library
react-redux	State management
react-router-dom	routing and link management

Server-side Dependencies	Purpose
express	Application framework
express-validator	validates requests received
puppeteer	Take screenshot of portfolio for thumbnail
mongoose	MongoDB management
config	read configuration file
axios	intercept requests
normalize-url	normalize urls inserted by user
tough-cookie	date parsing
cors	access resources from remote hosts

Architecture diagram
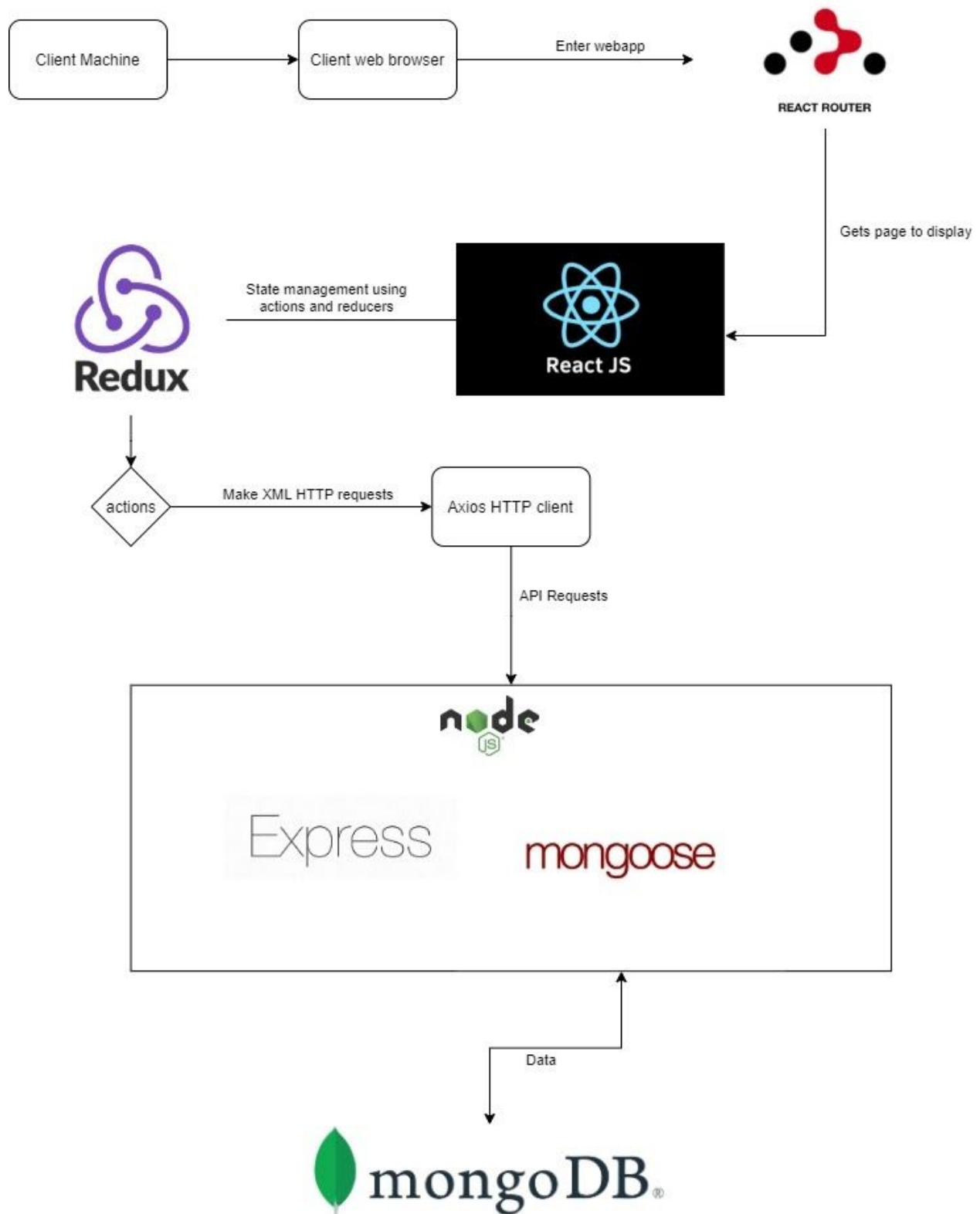
Post Sprint Team Discussion

What went well

- All members participated in sprint planning process
- Quick response time to issues/queries within the team, good cooperation
- Good system for validating pull requests
- Good use of trello for communication and information storage
- Well-established API provides an overview of project functionality
- Consistency of coding style maintained
- Team roles and responsibilities were assigned clearly
- Minimal code rewriting due to clearly defined project structure

What didn't go well and why

- Unable to complete all proposed requirements
  - Planned too many features during planning
- Mid sprint changes in functional requirements
  - Did not fully plan out features during planning
- Inefficient communication with client
  - Did not follow up within 3 days of sending email
- Documentation does not accurately reflect changes over time
  - Documentation was not incrementally updated during sprint
- Dependencies between work of teammates
  - Insufficient modularization/breakdown of tasks
- Insufficient confidence in error-free-ness of project
  - Did not manage to include automated testing in sprint

What can we change in the next sprint

- Schedule following meeting at the end of each meeting with clients
- Break down tasks more into specific and detailed parts
- Seek feedback from tutor more frequently
- Store version history of project documentation
- Assign all cards at start of sprint so that group members are well aware of their tasks

Screenshots

Github

Trello