CSCC01 eBeauty
System Design Document

Lim Lei,
Allen Chien-Luan Lo,
Jason Chen,
Jin Ting Zhou,
Junxun Liu,
Hyun Woo (Eddie) Shin,
Yuehai Zhou

July 30, 2022

Table of Contents

CRC Cards

Frontend Classes

Class: /frontend/src/index.js	
Responsibilities	**Collaborators**
• Used to render the app by combining all the components inside App.js into one final web app.	• App.js

Class: /frontend/src/App.js	
Responsibilities	**Collaborators**
• Manages the components of the app as a root component. • Used to render the app.	• /src/index.js

Class: /frontend/src/store.js	
Responsibilities	**Collaborators**
• A store that combines all the reducers in one root reducer	• index.js

Actions

Class: /frontend/src/actions/providerAction.js	
Responsibilities	**Collaborators**
• An action file contains two method to list all providers from backend • Fetch data from backend to frontend • Dispatch when needed	• /components/SearchPage.js

Authentication

Class: /frontend/src/Authentication/AuthProvider,js	
Responsibilities	**Collaborators**
• wrap and bind the context, make the context accessible for all the children under it	• index.js

Class: /frontend/src/Authentication/RequireAuth.js	
Responsibilities	**Collaborators**
• It's to restrict some pages to be only accessible for a certain identity, like only providers can access their own schedule, but customers are not able to view those pages.	• App.js

Class: /frontend/src/Authentication/useAuth.js	
Responsibilities	**Collaborators**
• basically a hook of context, every class can import it and access the context value easily. • critical use of it is to specify if logged in identity can access specific pages.	• /Authentication/RequireAuth.js • (any other class that want to access info of person logged in.)

Components

Class: /frontend/src/components/AddressForm.js	
Responsibilities	**Collaborators**
• Used to connect to create a form to handle user input for addresses.	• N/A

Class: /frontend/src/components/Alerting.js	
Responsibilities	**Collaborators**
• A helper class to replace alert(). looks more formal and user-friendly	• N/A

Class: /frontend/src/components/Animation.js	
Responsibilities	**Collaborators**
• animate the image inside findJon page	• findJob.js

Class: / frontend/src/components/CustomerCalendar.js	
Responsibilities	**Collaborators**
• the calendar that only the customer can access to, the customer • would only be able to click and do a reservation on an existing event. but not able to do any other modification.	• /pages/ServiceMenuPage.js

Class: / frontend/src/components/CustomerNavbar.js	
Responsibilities	**Collaborators**
• the Navbar user has after logged in.	• App.js

Class: / frontend/src/components/Footer.js

Responsibilities	Collaborators
• being footer of all pages	• server.js

Class: / frontend/src/components/Loader.js

Responsibilities	Collaborators
• A Loader when processing the data from DB, uses will see a spinner.	• /components/SearchPage.js • Could be used to other js files as well

Class: /frontend/src/components/MapContainer.js

Responsibilities	Collaborators
• Used to connect to google map API and display a map.	• N/A

Class: /frontend/src/components/Message.js

Responsibilities	Collaborators
• A message alters if errors appears	• /components/SearchPage.js

Class: /frontend/src/components/Navbar.js

Responsibilities	Collaborators
• Allows users to have access to homepage, register the different purpose's accounts and login	• N/A

Class: /frontend/src/components/Paginator.js

Responsibilities	Collaborators
• A pagination when too many data loaded in search page, users can navigate different page to see the results	• /components/SearchPage.js

Class: /frontend/src/components/ProviderCalendar.js

Responsibilities	Collaborators
• the calendar that only the provider can access to. • would be able to look at detail and add/delete an event when they want to do so.	• /components/ProviderScheduling.js

Class: /frontend/src/components/ProviderCard.js

Responsibilities	Collaborators
• List of task providers that show up in the search page when users search strings	• /components/SearchPage.js

Class: /frontend/src/components/ProviderNavbar.js

Responsibilities	Collaborators
• the Navbar Provider has after logged in.	• App.js

Class: /frontend/src/components/SampleGallery.js

Responsibilities	Collaborators
• Displays multiple images in ImageList form	• ViewProfileProvider.js

Class: /frontend/src/components/ScrollToTop.js

Responsibilities	Collaborators
• A helper class which resets the position of the client by setting the position of the viewer's location of the page to top after switching pages.	• N/A

Class: /frontend/src/components/SearchBox.js	
Responsibilities	**Collaborators**
• This is the search bar from search result page • Used to type strings	• /components/SearchPage.js

Class: / frontend/src/components/ServiceDropdown.js	
Responsibilities	**Collaborators**
• Allows users to specify the service they are searching for.	• /components/SearchPage.js

Constants

Class: / frontend/src/constants/countries.js	
Responsibilities	**Collaborators**
• Constant hashmap used to identify different countries.	• CheckoutAddressPage.js

Class: / frontend/src/constants/creeditCardmonths.js	
Responsibilities	**Collaborators**
• Constant hashmap used to identify months in credit card form.	• CheckoutPaymentPage.js

Class: / frontend/src/constants/creditCardYears.js	
Responsibilities	**Collaborators**
• Constant hashmap used to identify years in credit card form.	• CheckoutPaymentPage.js

Class: / frontend/src/constants/providerConstant.js	
Responsibilities	**Collaborators**
• store the provider string constant being used by other classes.	• /actions/providerAction.js • /reducers/providerReducer.js

Class: / frontend/src/constants/provinces.js	
Responsibilities	**Collaborators**
• Constant hashmap used to identify provinces in Canada for checkout.	• CheckoutAddressPage.js

Pages

Class: /frontend/src/pages/AddressPage.js	
Responsibilities	**Collaborators**
• Allows users to enter a new address or change address.	• MapContainer.js • Footer.js • AddressForm.js

Class: / frontend/src/pages/CheckoutAddressPage.js	
Responsibilities	**Collaborators**
• Customers provide their address information for service appointments	• N/A

Class: / frontend/src/pages/CheckoutPaymentPage.js	
Responsibilities	**Collaborators**
• Customers provide their credit card information for payment of the services they are trying to book.	• N/A

Class: / frontend/src/pages/CheckoutPaypalPage.js	
Responsibilities	**Collaborators**
• Customers are directly moved to checkout review page with paypal buttons. • Customers can make payment using their paypal account.	• CheckoutPaymentPage.js

Class: / frontend/src/pages/CheckoutReviewPage.js	
Responsibilities	**Collaborators**
• This page provides a summary overview of the orders that customers about to submit. • If all information is correct, the submitted information will be saved into DB.	• /backend/controllers/orderController.js • /backend/models/orderModel.js • /backend/server.js

Class: / frontend/src/pages/ContactPage.js	
Responsibilities	**Collaborators**
• Allows users to contact and email support when they need help/additional information.	• N/A

Class: / frontend/src/pages/FindJob.js	
Responsibilities	**Collaborators**
• Allows service providers to view the job page and access to their register page.	• /components/Animation.js

Class: /frontend/src/pages/HomePage.js	
Responsibilities	**Collaborators**
• Displays general information about the app. • Provides a quick search option. • Attracts potential users to sign up to the app.	• N/A

Class: / frontend/src/pages/Login.js	
Responsibilities	**Collaborators**
• Allows users and providers to login to their accounts. • Receives credential inputs for verification.	• /css/Login.css

Class: /frontend/src/pages/NotFound.js	
Responsibilities	**Collaborators**
• Used to redirect users back to the homepage when they encounter invalid URLs.	• N/A

Class: /frontend/src/pages/OnlyProviderView.js

Responsibilities	Collaborators
• test of context working, a page that only provider is accessible to.	• N/A

Class: / frontend/src/pages/OrderHistory.js

Responsibilities	Collaborators
• Allow users to view their past or upcoming orders	• /components/ReservationCustomer.js

Class: / frontend/src/pages/ProfileProvider.js

Responsibilities	Collaborators
• Allows service providers to edit their profile data. • Allows service providers to edit their account information.	• N/A

Class: / frontend/src/pages/ProviderScheduling.js

Responsibilities	Collaborators
• Allows the provider to schedule and manage their appointments	• N/A

Class: / frontend/src/pages/ReservtionCustomer.js

Responsibilities	Collaborators
• The file provides the customer to view a specific appointment/reservation they have made or already finished.	• App.js

Class: /frontend/src/pages/ReviewPage.js

Responsibilities	Collaborators
• Allows users to enter reviews with star ratings to their service providers	• App.js

Class: / frontend/src/pages/SearchPage.js

Responsibilities	Collaborators
• Displays a search result page after users provide search input • Allows users to search again and use filters with different attributes • Display user's address on file	• /components/ServiceDropdown.js • /components/ViewProfileProvider.js

Class: /frontend/src/pages/ServiceMenuPage.js

Responsibilities	Collaborators
• Provides available services of service providers for reservation to customers. • Provides available date and time in calendar form so that customers can choose the appointment time appropriately.	• /components/CustomerCalendar.js

Class: /frontend/src/pages/SignUp.js

Responsibilities	Collaborators
• Allows users to sign up/create new accounts for the app.	• /css/SignUp.css

Class: / frontend/src/pages/SignUpProvider.js

Responsibilities	Collaborators
• Allows service providers to fill in their information and register to the app as a first step. • Collect basic information about service providers.	• N/A

Class: /frontend/src/pages/SignUpProviderTwo.js

Responsibilities	Collaborators
• Allows service providers to view the job page and access their registration page as a second step. • Collects additional information about service providers.	• N/A

Class: /frontend/src/pages/SignUpProviderThree.js	
Responsibilities	**Collaborators**
• Allows service providers to upload their profile image and access their registration page as a third step. • Collects additional information about service providers.	• N/A

Class: / frontend/src/pages/Unauthorized.js	
Responsibilities	**Collaborators**
• supposed to show if a person wants to access pages that his/her identity does not allow. • currently has not been implemented yet.	• N/A

Class: / frontend/src/pages/ViewProfileProvider.js	
Responsibilities	**Collaborators**
• Allows users to view the profiles of service providers.	• N/A

Reducers

Class: /frontend/src/reducers/providerReducer.js	
Responsibilities	**Collaborators**
• Request reducer for all providers • Request reducer for a provider detail	• store.js

Backend Classes

Class: /backend/server.js	
Responsibilities	**Collaborators**
• settle express, defines endpoints and manages API methods.	• NA

Class: /backend/seeder.js	
Responsibilities	**Collaborators**
• load/delete sample to database	• NA

Config

Class: /backend/config/db.js	
Responsibilities	**Collaborators**
• connect server to database, need .env to provide connection url	• server.js • seeder.js

Controllers

Class: /backend/controllers/calendarController.js	
Responsibilities	**Collaborators**
• create a new timeslot (of calendar) in calendar DB • There are 4 retrieve methods • GET all calendars • GET all calendars from a provider • GET detail from start/end time & providerID • PATCH timeslot with customer ID • DELETE timeslot	• Server.js • /models/calendarModel.js

Class: /backend/controllers/customerController.js

Responsibilities	Collaborators
• Registers a new customer by creating a new entry and updating it in Customer DB • Allows customers log into the app by checking their login credentials with credentials stored in Customer DB • Retrieves a single customer or entire customers data from DB • Deletes a customer entry from DB • Updates attributes of customer entry in DB • Retrieves the customer's address data from the DB • Updates the user's address data in the DB	• Server.js • /models/customerModel.js

Class: /backend/controllers/issueController.js

Responsibilities	Collaborators
• manage posting issue to DB	• Server.js • /models/issueModel.js

Class: /backend/controllers/orderController.js

Responsibilities	Collaborators
• This page provides a summary overview of the orders that customers are about to submit. • If all information are correct, the submitted information will be saved into DB.	• /backend/controllers/orderController.js • /backend/models/orderModel.js • /backend/server.js

Class: /backend/controllers/postController.js

Responsibilities	Collaborators
• Manages post entries of the app • Retrieves post entries from DB • Creates/Inserts a new post entry into DB • Deletes a post entry from DB • Updates a post entry in DB	• Server.js • /models/postModel.js

Class: /backend/controllers/providerController.js	
Responsibilities	**Collaborators**
• Registers a new service provider by creating a new entry and updating it in DB • Allows service providers log into the app by checking their login credentials with credentials stored in DB • Retrieves a single provider by Id or entire providers data from DB	• Server.js • /models/providerModel.js

Class: /backend/controllers/reviewController.js	
Responsibilities	**Collaborators**
• Manages review entries of the app • Retrieves review entries from DB • Creates/Inserts a new review into DB • Deletes a review entry from DB • Updates a review entry in DB	• Server.js • /models/reviewModel.js

Data

Class: /backend/data/metrics.js	
Responsibilities	**Collaborators**
• sample data of provider data dashboard	• seeder.js

Class: /backend/data/order.js	
Responsibilities	**Collaborators**
• sample data of order database	• seeder.js

Class: /backend/data/post.js	
Responsibilities	**Collaborators**
• sample data of post database	• seeder.js

Class: /backend/data/profile.js	
Responsibilities	**Collaborators**
• sample data of profile database	• seeder.js

Class: /backend/data/provider.js	
Responsibilities	**Collaborators**
• sample data of provider database	• seeder.js

Class: /backend/data/review.js	
Responsibilities	**Collaborators**
• sample data of review database	• seeder.js

Class: /backend/data/metrics.json	
Responsibilities	**Collaborators**
• sample data of provider data dashboard	• ProfileProvider.js

Class: /backend/data/order.json	
Responsibilities	**Collaborators**
• sample data of provider data dashboard	• ProfileProvider.js

Class: /backend/data/post.json	
Responsibilities	**Collaborators**
• sample data of provider data dashboard	• ProfileProvider.js

Middleware

Class: /backend/middleware/authMiddleware.js	
Responsibilities	**Collaborators**
• Checks authentication for access	• Server.js • /models/providerModel.js

Class: /backend/middleware/errorMiddleware.js	
Responsibilities	**Collaborators**
• Handles NotFound and DB query errors	• Server.js

Class: /backend/middleware/upload.js	
Responsibilities	**Collaborators**
• create a bucket for uploading images. (GRIDFS)	• /routes/imageRoute.js

Models

Class: /backend/controllers/calendarModel.js	
Responsibilities	**Collaborators**
• Defines MongoDB schema for Calendar DB	• Server.js • /models/calendarController.js

Class: /backend/models/customerModel.js	
Responsibilities	**Collaborators**
• Defines MongoDB schema for Customer DB	• Server.js • /controllers/customerController.js

Class: /backend/controllers/issueModel.js	
Responsibilities	**Collaborators**
• Defines MongoDB schema for issueDB	• Server.js • /controllers/issueController.js

Class: /backend/controllers/orderModel.js	
Responsibilities	**Collaborators**
• Defines MongoDB schema for order collection	• Server.js • /controllers/orderController.js

Class: /backend/models/postModel.js	
Responsibilities	**Collaborators**
• Defines MongoDB schema for Post DB	• Server.js • /controllers/postController.js

Class: /backend/models/providerModel.js	
Responsibilities	**Collaborators**
• Defines MongoDB schema for Provider DB	• Server.js • /controllers/providerController.js

Class: /backend/models/reviewModel.js	
Responsibilities	**Collaborators**
• Defines MongoDB schema for Review DB	• Server.js • /controllers/reviewController.js

Class: /backend/models/serviceModel.js	
Responsibilities	**Collaborators**
• Defines MongoDB schema for Service DB	• Server.js

Routes

Class: /backend/controllers/calendarRoute.js	
Responsibilities	**Collaborators**
• Defines the endpoints related to /api/calendar route using express Router.	• Server.js • /models/calendarController.js

Class: /backend/routes/customerRoute.js

Responsibilities	Collaborators
• Defines endpoints for Customer DB • Connects controllers through router	• Server.js • /controllers/customerController.js

Class: /backend/routes/imageRoute.js

Responsibilities	Collaborators
• Defines endpoints for image DB • does the connection of database on it's own • post image, get image, and delete image from DB	• Server.js

Class: /backend/controllers/issueRoute.js

Responsibilities	Collaborators
• Defines endpoints for issueDB • Connects controllers through router	• Server.js • /controllers/issueController.js

Class: /backend/controllers/orderRoute.js

Responsibilities	Collaborators
• Defines the endpoints related or /api/orders route using express Router.	• Server.js • /controllers/orderController.js

Class: /backend/routes/postRoute.js

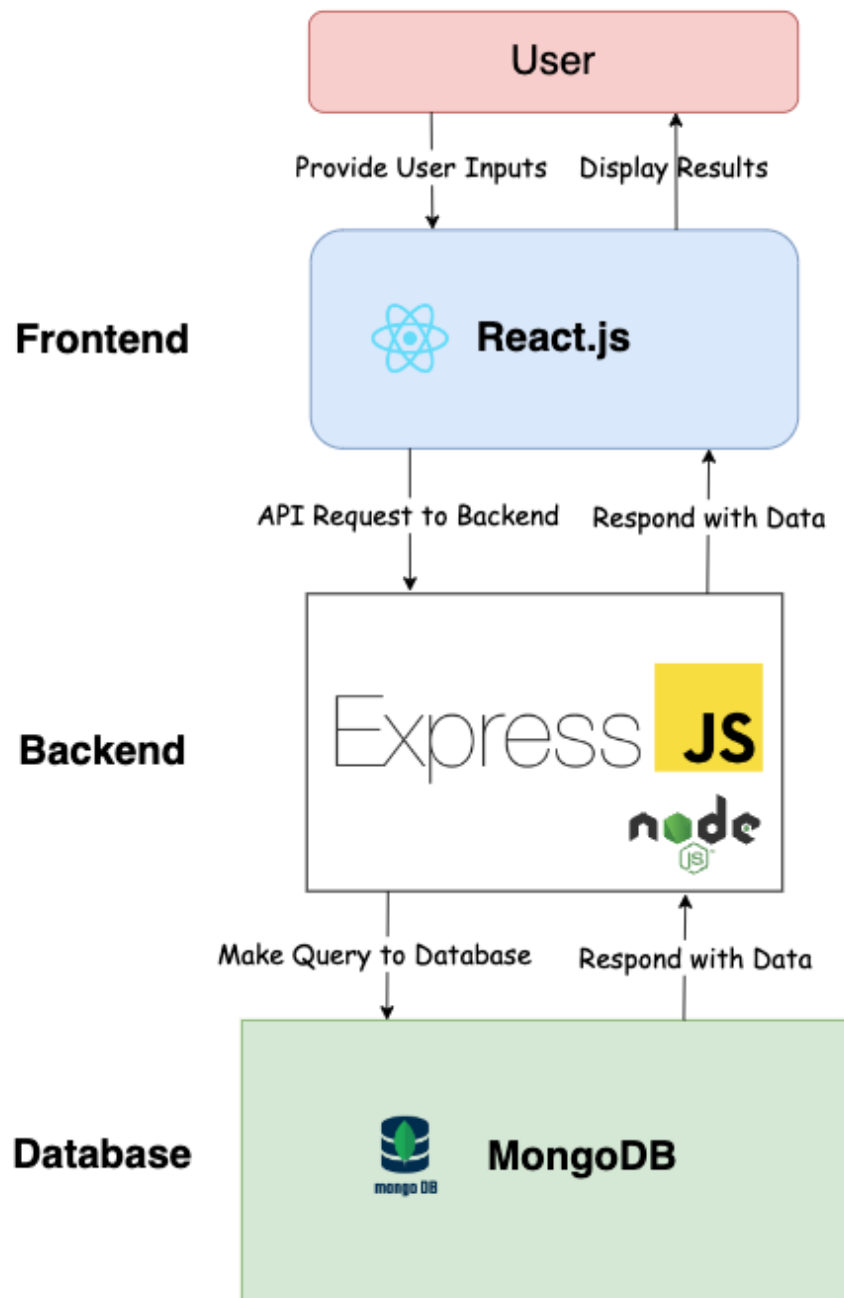| Responsibilities	Collaborators
• Defines endpoints for Post DB • Connects controllers through router	• Server.js • /controllers/postController.js

Class: /backend/routes/providerRoute.js

Responsibilities	Collaborators
• Defines endpoints for Provider DB • Connects controllers through router	• Server.js • /controllers/providerController.js

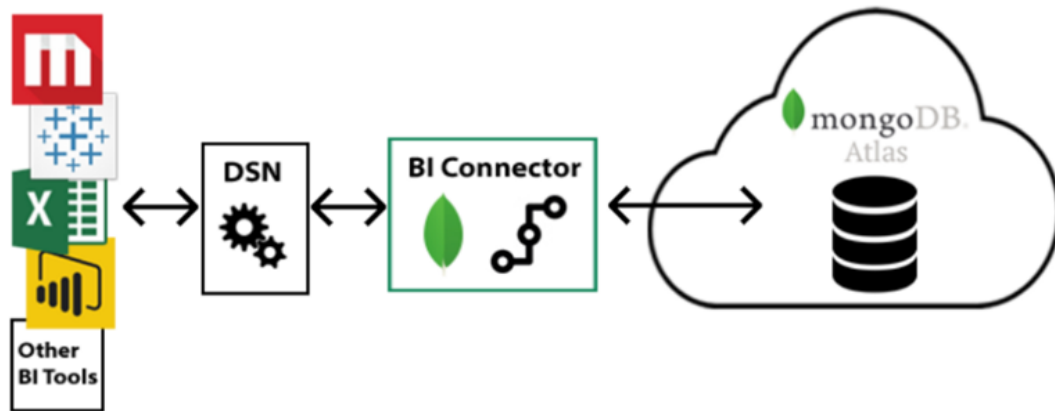| Class: /backend/routes/reviewRoute.js	
Responsibilities	**Collaborators**
• Defines endpoints for Review DB • Connects controllers through router	• Server.js • /controllers/reviewController.js

System Architecture Design

MERN Stack

Our web application follows a 3-tier architecture which is composed of frontend, backend, and database using JavaScript and JSON. React.js is responsible for the frontend operations of the app while Express.js and Node.js perform server-side operations. To store the data from the app, we use MongoDB, a document-oriented NoSQL database. This 3-tier architecture is also known as MERN stack which is an abbreviated form of **M**ongoDB, **E**xpress, **R**eact, and **N**ode.

Mongo Connect Others

System Flow Decomposition

Users first interact with the UI components of React.js in the web browser. They can browse the different pages of the app and provide inputs for different operations such as login/register, search, comments, or reviews. Since the web page is served by Express.js running on Node.js, whenever users provide inputs, the server-side applications can collect them and send them to the MongoDB database. Also, whenever users request data from the database, Express.js is responsible for retrieving the data from the database through querying.