ASSIGNMENT 7 REPORT

Group: 03

Misha Herasimov (Sub-team 1)
Sneha Jayavardhini Doguparthi (Sub-team 1)
Yuganthi Krishnamurthy (Sub-team 2)
Ruize Nie (Sub-team 2)
Ram Prasath Meganathan (Sub-team 3)
Pratheep Kumar Manoharan (Sub-team 3)
Pratheep Kumar Manoharan (Sub-team 4)
Ram Prasath Meganathan (Sub-team 4)

Table of Contents

Figures

Tables

Content

The current assignment submission contains a list of next documents:

- A7-Team03-Minutes.pdf – The summary of all meeting conducted by the team
- A7-Team03-TasksForm.pdf – The form that contains a description of major tasks completed
- A7-Team03-Report.pdf – The details description of the assignment
- A7-Team03-Code.zip – The archive with the implantation code for three companies
- A7-Team03-Db.zip – The SQL scripts that generate the database and its initial content

Deficiencies & Extra Work

The assignment satisfies all the requirements. We implemented the infrastructure for three companies that interact with each other using public APIs. We excelled at initial requirements by using two cloud providers for serverless computing. The company X backend uses Lambda in AWS while company Y uses Google Cloud Functions. Moreover, the frontends for both company X and Y were deployed in AWS S3 buckets as a static webpage. This means that the infrastructure for the two companies is completely serverless. The backend of company X and Y are deployed in serverless functions in AWS Lambda and Google functions. To satisfy requirements we deployed company Z infrastructure using Docker in a fully managed AWS Elastic Bean Stalk. The companies' data are stored in the AWS Relational Database Service. We reused the previously created database. The user interface for three websites was implanted using both React library and Angular frameworks. The interface was configured with the use of the Bootstrap library. The interface for company Z provided a reach error handling; the users will be immediately notified in case of an error with visual description.

Trello Board

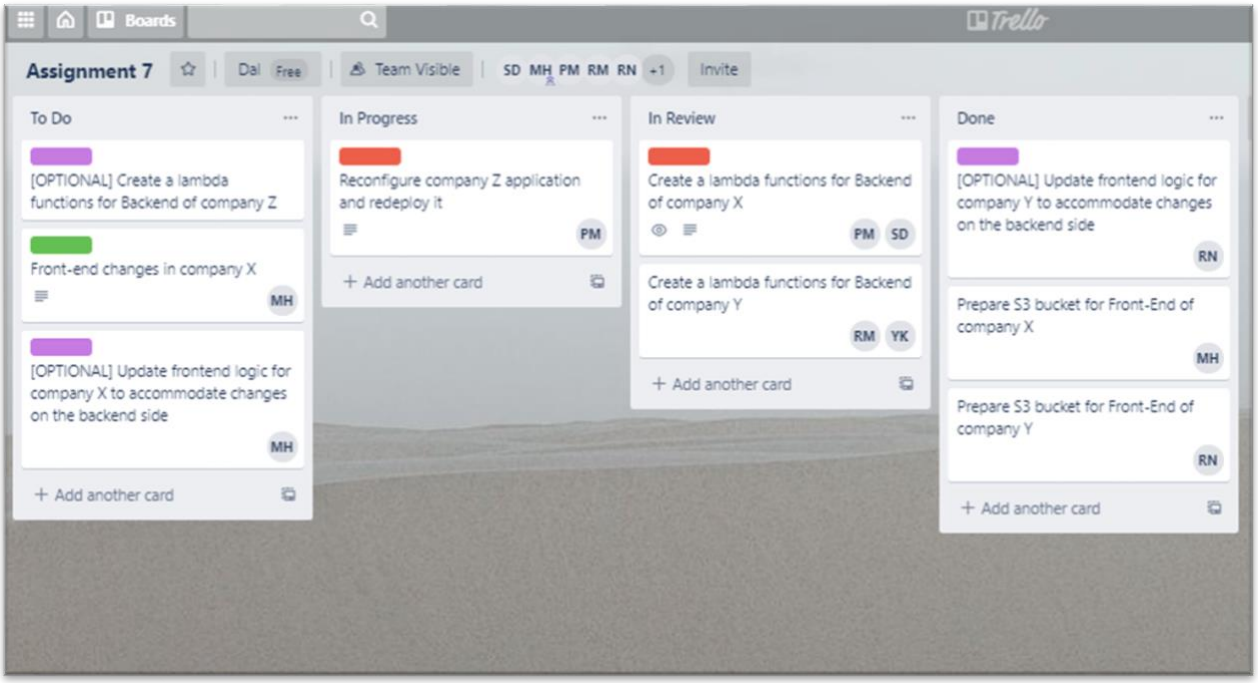

Figure 1. Assignment VII Trello board

Fig. 1 shows our Trello board [1] where all the cards represent different tasks. We have added four lists i.e. ‘To Do’, ‘In Progress’, ‘In Review’ and ‘Done’. These lists helped us to easily track the progress of each card/task of all the team members. After creating the lists, the tasks were created and the requirement for each task is identified and written as part of the descriptions in each card. Fig. 2 shows the details for creating a lambda function for Company X backend and Fig. 3 shows the details about reconfiguring Company Z application and redeploying it.

Figure 2. Example of task created for the company X

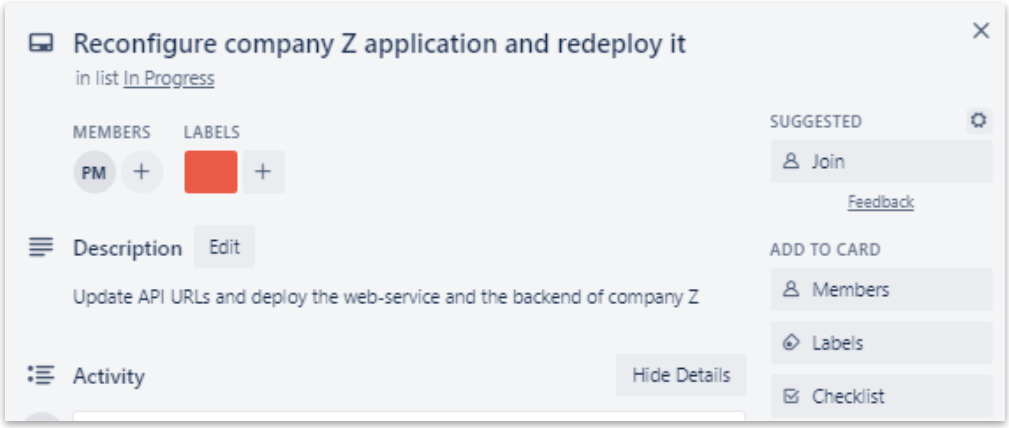

Figure 3. Example of task created for the company Z

Software organization

Company X

The infrastructure for the company X was created using Express.js as the backend library and Angular for implementation of web interface [2]. Using a web portal, the users can make all the CRUD manipulation with job data. The company X backend communicates with the AWS RDS to store Job content. To preserve data integrity, the backend contacts company Y public API to fetch the most recent list of Parts.

The API for a company X consists of nine endpoints that are utilized by the company X web interface as well as the company Z backend. The backend infrastructure was reimplemented for serverless computing. The existing code was used as the backbone for newly created Lambda functions. The functions itself were uploaded to the AWS and the additional infrastructure that handles API Gateway was configured. As for the visual part of the company X, the endpoints and response handling were updated to support serverless backend. The Angular application was locally built and uploaded to the S3 bucket. The bucket storage was configured accordingly to support static web page hosting [3, 4].

The interface of company X is demonstrated in Fig. 4, 5, 6. The first figure illustrates the currently present list of jobs. By pressing the buttons at each row, the user can select the type of manipulation with content and delete or update existing job information. On the right side of the page, the user can compose and create a new job.

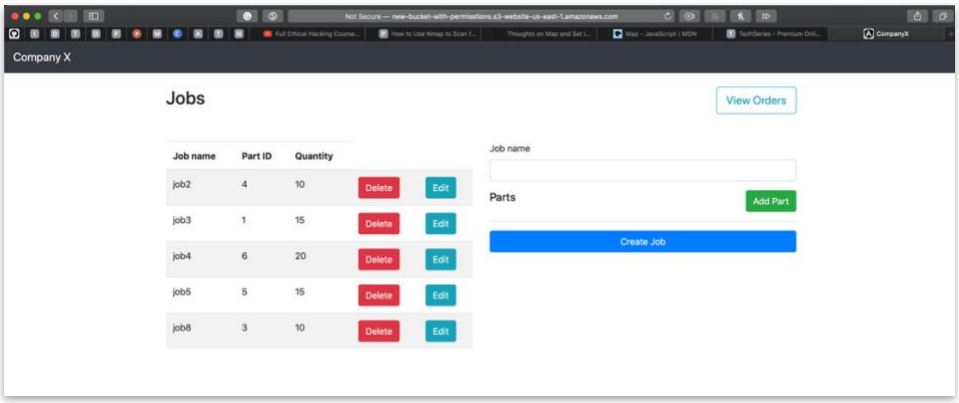

Figure 4. Company X main page

Fig. 5 shows a list of orders places using company X web interface.

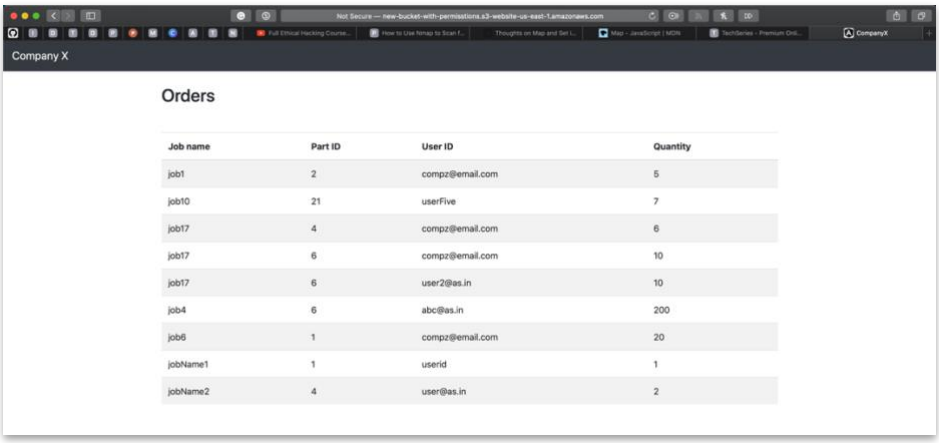

Figure 5. List of order items page

Fig. 6 shows a page where user can edit individual job. The name of the job and its parts are grayed to prevent users from editing these fields.

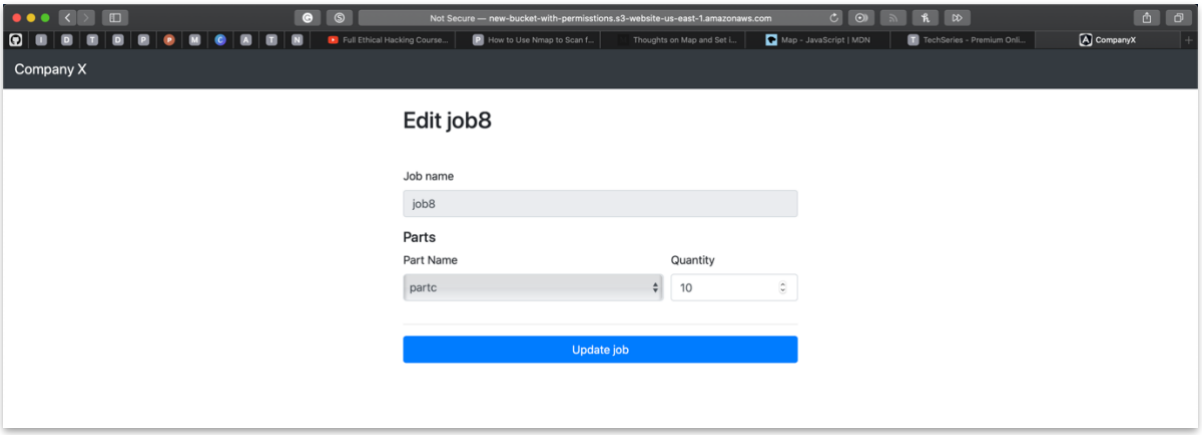

Figure 6. Edit job page

Company Y

The company Y front end was built using React framework and. Node.js and Express.js was used for the backend of company Y [5, 6]. The AWS RDS (MySQL) was used to handle the data [11, 12]. The frontend of company Y is deployed on AWS S3 bucket.

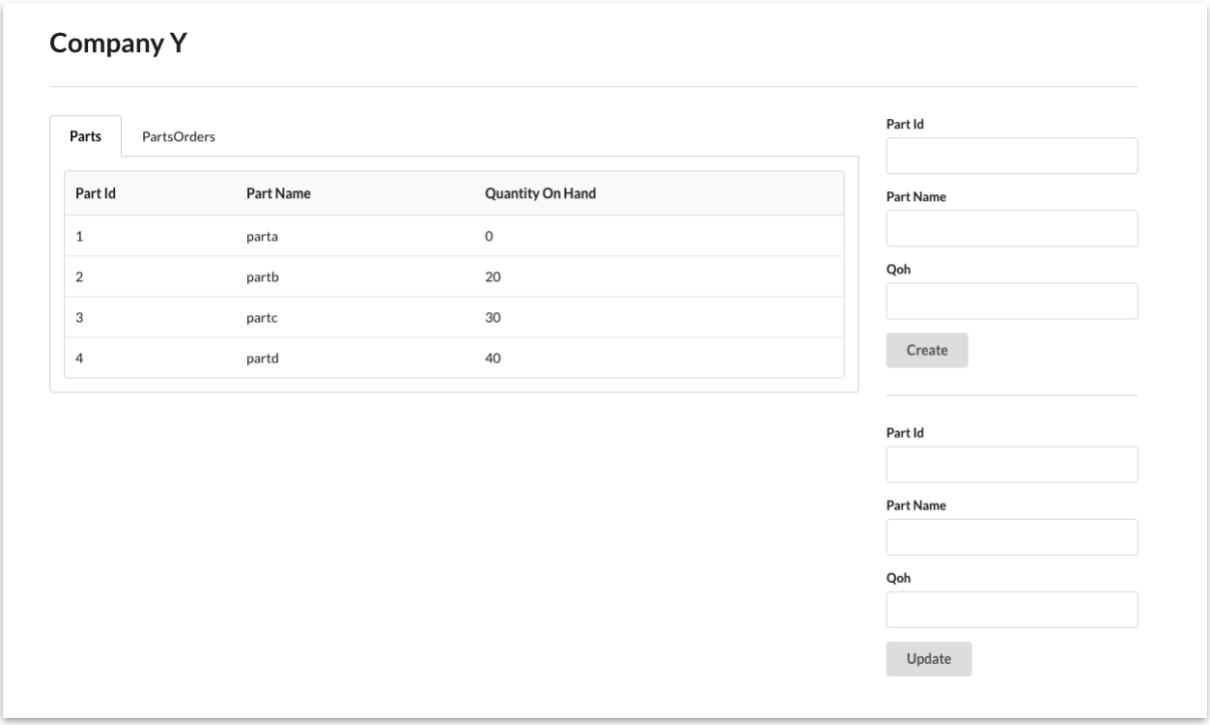

Figure 7. The front-end UI of Company Y

Company Y provides the CUR operation for Parts. The endpoints for company Y have been configured so that the company Z can inform company Y to update the part order table when there are new orders coming in. The company X can also use the company Y endpoint to retrieve the part with part ID [11,

12]. The backed is built as express service with database as MySQL and deployed in Google Cloud as a serverless cloud function.

Table 1. Libraries used for implementation of company Y Backend

№	Library	Purpose
1	Express.js	Express js is node js package provided for creating endpoints to perform CRUD operations in the database [6].
2	mysql	MySQL package is used in Express.js to establish a connection with the database [7]
3	Firebase-functions	This package is used to deploy the backend of Company Y to Google Cloud as a serverless function [2]

Company Z

The front-end and back-end of the company Z are created as two different modules. The communication between the modules are done using the REST API calls in a secure way. The front-end of the application is developed using ReactJS framework. Since this is an extension of Assignment 6, Have added reference to assignment 6 as [11, 12]**.**

Table 2. Libraries used for implementation of company Z frontend [5]

№	**Library**	**Purpose**
1	axios	Axios is a promise-based HTTP client used to send asynchronous HTTP requests to backend endpoints from frontend.
2	react	React framework library used to render the component and update the state.
3	react-bootstrap	Styling library is used to add styles, effects, and effectiveness.
4	react-dom	The react-dom package is used at the top level of the app and as an escape hatch to get outside of the React model.
5	react-router-dom	The routing between pages, history maintenance, parent and child page relations are managed using this library.
6	react-scripts	The build and deployment of the app is done using this library.

The front-end user interacting pages of the company Z are added below. Fig. 10 shows the landing page of the company Z application. We have a short description in the landing page taken from the assignment requirement document. The landing page has a navigation bar that can take the users to Search page and list all jobs page. The two pages has a REST API calls to company X to fetch the job details. The

jobname from company X will be listed in the screen. The user must click on the jobname that needs to be ordered.

All the activities mentioned above can be done without authentication. Any user with application URL can do. On click of jobname in Search page or Get All Jobs page, the application will take the user to the login page. In the login page, the user must provide valid username and password to proceed. The valid user credentials will take the user to the homepage of the application. In the homepage, we have the order page and search history page. The order page contains all the details about the job, parts, available quantity, and required quantity. The search history page shows the recent searches done all users in the application. The order page fetch part details from the company Y. The user must select the parts that they want to order and click order button. This will trigger three APIs one to each company's backend to store the order details.

Fig. 8 shows the architecture of company Z application. The drawing is created using edraw max tool online [8].

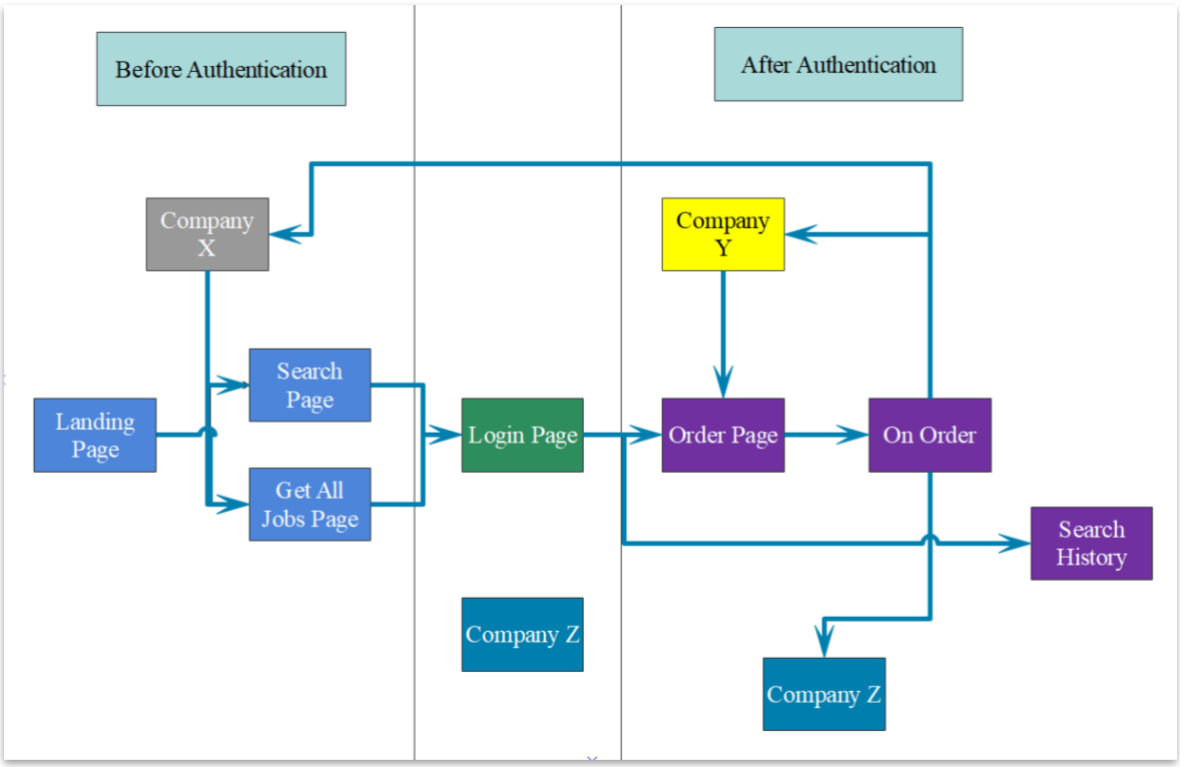

Figure 8. Architecture of Company Z [2]

Fig. 9 shows the landing page of the application. This is the first page that user interacts with the application.

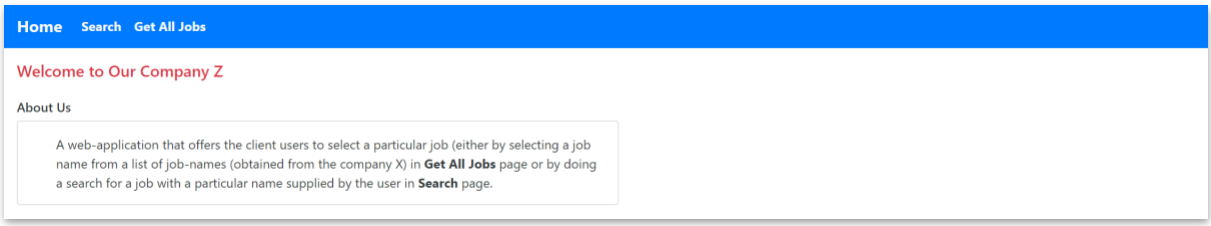

Figure 9. Landing page of Company Z

Fig. 10 shows the search page of the application. The user must enter the jobname and click on search to fetch the details from company X. The search will work even if the user tries with part of jobname.

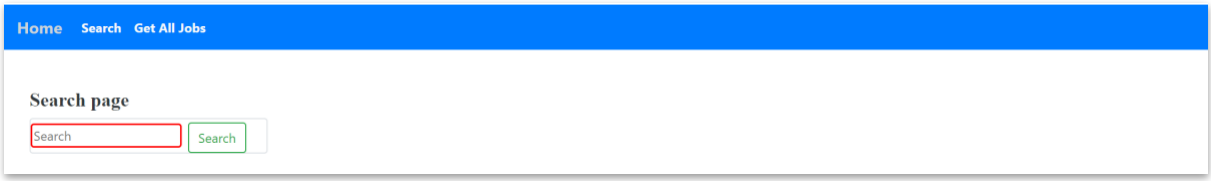

Figure 10. Search page of Company Z

Fig. 11 shows all jobnames from the company X. The user must click on the required jobname and this will take the user to login page.

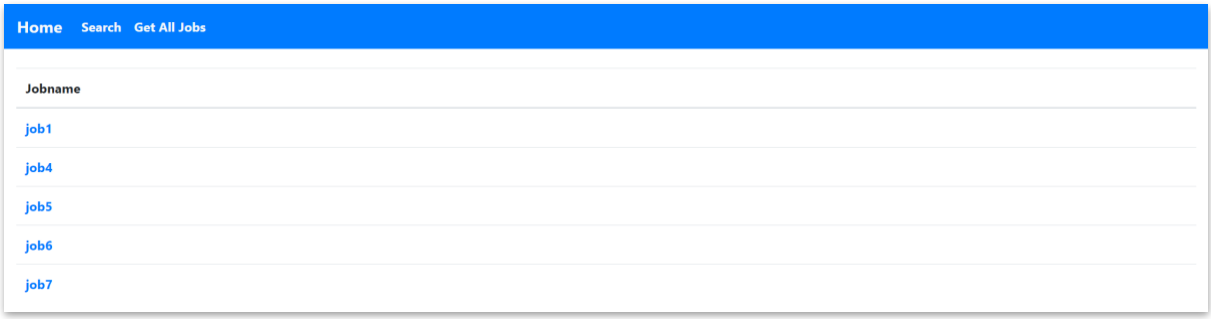

Figure 11. Get all jobs page of Company Z

Fig. 12 shows the login page where the application will authenticate the user.

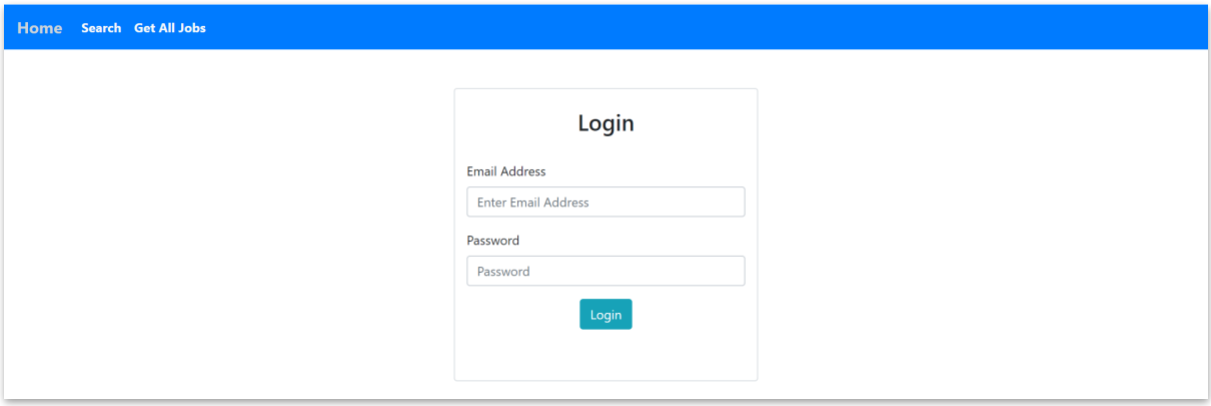

Figure 12. Login page of Company Z

Fig. 13 shows the Order page of the application. The order page fetches the job details from company X and part details from company Y. The order page has below logics built in.

1. One user cannot order the part that was already ordered.
2. User can order any number of parts.
3. The order details will be posted to all three companies.
4. The minimum frontend validations like required quantity and available quantity comparison are done.

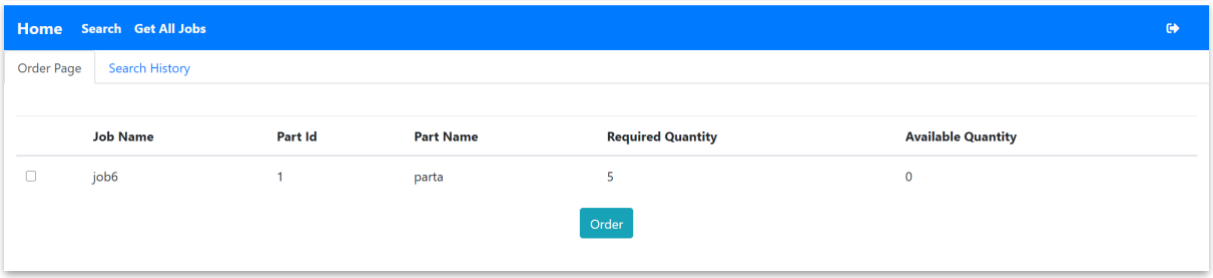

Figure 13. Order page of Company Z

The backend of the application is developed using Express.js framework which is used to create the server and API requests to be called from front end. Node MySQL library is used to connect the Node.js backend to the MySQL instance of the AWS RDS database. Following are the list of libraries used in the backend of the application.

Table 3. Libraries used for implementation of company Z backend

№	Library	Purpose
1	Express.js	Express js is node js package provided for creating endpoints to perform CRUD operations in the database [6].
2	mysql	MySQL package is used in Express.js to establish a connection with the database [7].
3	jsonwebtoken	This library is used to generate the authentication token for providing to the frontend for authenticating and maintaining the user session [9].
4	body-parser	Body-parser is used to read the body of the request and convert it to json for further operations [10].
5	Firebase-functions	This package is used to deploy the backend of Company Y to google cloud as a serverless function [2].

Company Z uses three tables mainly to perform CRUD operations

- Search – used to store the search history of the user
- User - stores the credentials of the user for authentication

- JobParts - stores the details of the JobParts that are ordered through the application

Access Information

The code for all three companies is stored in single Gitlab repository. The URL is provided below:

https://git.cs.dal.ca/herasimov/a7-team03

There was an individual folder created for each company. The README files contain information about the references and usage of 3-rd party code, if used at all.

References

[1] "Trello", *Trello.com*, 2020. [Online]. Available: https://trello.com/b/JGYHYysb/assignment-6. [Accessed: 13- Jul- 2020].

[2] *Angular*. [Online]. Available: https://angular.io/docs. [Accessed: 13-Jul-2020].

[3] “Cloud Functions | Google Cloud,” *Google Cloud*, 2020. [Online]. Available: https://cloud.google.com/functions. [Accessed: 28-Jul-2020]

[4]"AWS Lambda – Serverless Compute - Amazon Web Services", Amazon Web Services, Inc., 2020. [Online]. Available: https://aws.amazon.com/lambda/. [Accessed: 28- Jul- 2020]

[5] “Semantic UI React The official Semantic-UI-React integration.,” *Introduction - Semantic UI React*. [Online]. Available: https://react.semantic-ui.com/. [Accessed: 13-Jul-2020].

[6] ”Express - Node.js web application framework,” *Expressjs.com*, 2017. [Online]. Available: https://expressjs.com/. [Accessed: 12-Jul-2020]

[7] “mysql,” *npm*, 23-Jan-2020. [Online]. Available: https://www.npmjs.com/package/mysql. [Accessed: 12-Jul-2020]

[8] “Edraw Max, All-in-One Diagram Software.” Edrawsoft, www.edrawsoft.com/edraw-max/. [Accessed: 13-Jul-2020].

[9] “jsonwebtoken,” *npm*, 18-Mar-2019. [Online]. Available: https://www.npmjs.com/package/jsonwebtoken. [Accessed: 12-Jul-2020]

[10] “body-parser,” *npm*, 25-Apr-2019. [Online]. Available: https://www.npmjs.com/package/body-parser. [Accessed: 12-Jul-2020]

[11] “Submission History - CSCI4145 & CSCI5409 - Cloud Computing (Sec 1) - 2020 Summer - Dalhousie University,” *Brightspace.com*, 2020. [Online]. Available: https://dal.brightspace.com/d2l/lms/dropbox/user/folders_history.d2l?db=75915&grpid=0&isprv=0&bp=0&ou=124054. [Accessed: 12-Jul-2020]

[12]- “Submission History - CSCI4145 & CSCI5409 - Cloud Computing (Sec 1) - 2020 Summer - Dalhousie University,” *Brightspace.com*, 2020. [Online]. Available: https://dal.brightspace.com/d2l/lms/dropbox/user/folders_history.d2l?db=77200&grpid=128091&isprv=0&bp=0&ou=124054. [Accessed: 28-Jul-2020]