Problem Statement:(40 points)

Build, deploy and demo a simple banking app on the cloud. The app should consist of,

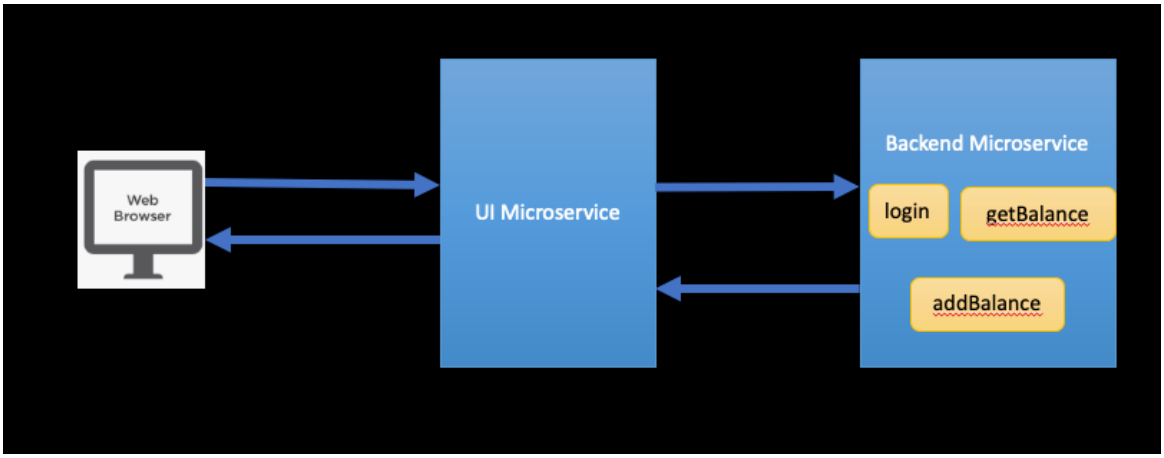
a) Front-end microservice accessible through any browser. The front-end microservice will be developed using NodeJS, HTML, CSS. It should show a simple Web page that will let user log-in, check account balance.

b) Backend microservice accessible via APIs. This microservice should be developed using NodeJS and will be consumed by the frontend microservice. The backend microservice will offer set of APIs to log the user in and check the account balance.

Tools & Pre-requisites

1. An active IBM Cloud account
2. Any IDE such as Visual Studio Code or IBM Cloud Eclipse editor
3. Basic knowledge of Node.js/Javascript
4. Basic knowledge of HTML, JavaScript & CSS
5. Basic knowledge of Postman tool to test end points

Overall architecture



What to Submit

1. The two micro service code and the serverless code
2. Screenshot of login/test results page
3. Sample Postman output of backend microservice API
4. Viva
5. Show the three optional steps and claim additional bonus points

Step 1 : Build and deploy the back end service on IBM Cloud

Create a Node.js application and deploy it on IBM Cloud. The reference documentation will help you get started.

This application must expose two REST endpoints

- /getBalance GET - To return the balance of the current user's wallet
- /addFunds PUT - Add the specified amount to the wallet.

For simplicity, the account balance of the user(s) can be held in memory within the application.

(Optional for Bonus pts only): Store and fetch the balance from storage service (e.g. Cloudant) and update the same.

Test your application's REST endpoints using Postman (https://www.postman.com/downloads/) or cURL

References:

https://cloud.ibm.com/docs/node

https://github.com/IBM-Cloud/cf-nodejs-c2c-demo

https://developer.ibm.com/languages/node-js/tutorials/learn-nodejs-node-with-cloudant-dbaas/

https://learning.postman.com/docs/getting-started/introduction/

Step 2 : Build and deploy the front-end microservice

Create a Node.js application that delivers a front-end UI to display the balance and add funds to the wallet. The user interface needs to be built with HTML, JavaScript and CSS. Call the backend micro service (milestone 1) to fetch the wallet balance and to add funds to the wallet.

(Optional for Bonus pts only): Use a framework such as React.js, Vue.js to build the app.

References:

https://medium.com/weekly-webtips/create-and-deploy-your-first-react-web-app-with-a-node-js-backend-ec622e0328d7

Milestone 3 : Secure your applications

- Secure the backend application - Secure your backend REST APIs by means of an API key or username/password. Test the same using Postman
- Secure the frontend application - Secure the frontend application by introducing a login page and authenticating the user credentials. Only allow authenticated users to view/update the balance & funds page.

(Optional for Bonus pts only): Use any 3rd party OAuth provider to secure your application. (e.g. Google, App ID on IBM Cloud)

Milestone 4 : Create an instance of IBM Cloud Functions and define an action to redirect page

(e.g. your app's v2. For now, redirect to google.com)

Create an instance of IBM Cloud Functions and define an action which will redirect the request to google.com. The action defined on IBM Cloud functions should accept the request redirect the page.

Test your function by manually triggering the action from the IBM Cloud console and ensuring the request is redirect.

References :
https://cloud.ibm.com/docs/openwhisk?topic=openwhisk-getting-started
https://cloud.ibm.com/docs/openwhisk?topic=openwhisk-actions_web
https://cloud.ibm.com/docs/openwhisk?topic=openwhisk-actions_web#http_redirect

Grading for Assignment 1 (40 points)

- Each step deliverable = code, screenshots, API testing
- 4 Steps x 8 points = 32 points
- Overall Viva = 8 points
- 3 Bonus Steps x 2 = 6 points