**RevSpeed - 1**

Application Overview

RevSpeed is an online account management application for internet and broadband service provider customers. It allows new customers to easily sign up for services and existing customers to manage their accounts. It provides features like account registration and login, user profile management, browsing the internet and broadband plans and subscription management.

Core Functional Scope

As a user, I want to:

1. Register and create an account to access the services.
2. Login to my account to access and manage my services.
3. Create a basic profile with name, phone number, email, and address so my account is personalized.
4. Update my personal details so my profile is accurate.
5. Easily view internet and broadband plans so I can select the best option for me.
6. View my internet and broadband subscriptions so that I can keep track of my subscriptions.
7. Apply for a new internet and broadband plan that is best suited for me.
8. Opt out of a subscribed plan if I no longer require it.
9. Make monthly/quarterly/annual payments. – *Nice to Have*
10. Receive notifications about my account so that I am promptly informed of changes or issues (Payment due dates, changes to subscription plans, and surge outages). – *Nice to Have*

Standard Functional Scope

Registered users should be able to log in, change the password, and request for a forgotten password (will be sent to their registered email).

Definition of Done

* Working application demonstration.
* Sharing the associates’ code repo for technical evaluation with:
* ERD Diagram
* Architecture Diagram

Technologies to be used.

HTTP/CSS, JavaScript, TypeScript, Angular, Spring Core, Spring Boot, SpringWeb, Spring Data JPA

**Expectations**

**Web Fundamentals (HTML, CSS and JS)**

**User Experience:**

1. Have an intuitive design for the user to work with the application without any training or guidance
2. Have clean & consistent UI, color theme and easy to use navigations
3. Use bootstrap framework for responsive pages
4. Have proper tab indexing for users to navigate between the fields without usage of mouse.

**User Inputs & outputs:**

1. Have appropriate HTML fields for the user inputs
2. Wherever possible use the client-side validations for the user input
3. Display the appropriate user info/error message with appropriate colors and icons

**Performance:**

1. Use compressed images / assets to increase the page performance
2. Use the application validated using the Chrome’s Lighthouse tool and improved based on the report

**Dataset:**

1. For any prepopulated data such as to render table rows use JSON file as the DataSource and use standard open-source library to read and render in the web pages.

**General standards:**

1. Ensure the w3 standards are implemented for better accessibility. E.g., Using alt attribute for image tag.
2. Ensure the SEO recommended meta tags are added.

**Web Development with Angular**

**Framework Specific**

1. Ensure the appropriate APIs are used for any of the API calls
2. Ensure the routing is centrally configured
3. Best practices & design patterns are to be followed
4. Implement the end-to-end testing framework and get to know the headless execution of end-to-end framework.

**Deployment artifacts:**

1. The deployment artifacts should be minified and obfuscated if required.

**Security**:

1. Ensure the CORS restriction is applied, if applicable.
2. Ensure Route Guarding/Authenticated Routing is implemented.
3. Ensure that the secrets are stored as environment variables using secure credential storage.

**Spring Boot**

**REST Standards:**

1. Ensure the REST standards are followed for API naming, HTTP Operation and Response (output definition)
2. Secure the protected APIs
3. Define a common URL pattern for public and secure APIs
4. Proper documentation of APIs with Input and Output Samples to be documented
5. Ensure that each micro service is defined to do one job.
6. Provide / enable to API gateway to route the request through a single channel
7. Ensure the micro services are enabled for traceability and monitoring with appropriate dashboard to render the micro services health and logging.
8. Use the recommended design patterns to aggregate the micro services (wherever is applicable)

**Modularity:**

1. If the application is enhanced from an existing console-based project, try to reuse the existing modules.
2. Refactor the existing code based on the current tech stacks

**Logging:**

1. Ensure the application is using proper logging framework and methods.
2. Ensure the application’s log level is configured using configuration files so that it can be changed without changing the code.
3. Also ensure that the application logging is configured to output to the mentioned log file.
4. Ensure the centralized logging implementation

**Testing**:

1. Ensure sufficient test cases are written using appropriate testing frameworks.
2. Ensure the code coverage closed to be 80%

**Security**:

1. Ensure the CORS restriction is applied.
2. SQL Injection thread is taken care
3. Throttling is to be taken care of.
4. Cross site Scripting to be avoided.
5. Ensure that the secrets are stored as environment variables using configuration files or secure credential storage.