

BlanceBuddy - LifestyleApp

Camelia Morari

July 2024

Contents

1	Introduction	3
1.1	Application Description	3
1.2	Used Technologies	3
1.3	Functionalities	3
1.4	User Stories	4
1.5	Interfaces, Module Communication and Protocol	6
1.6	Message Flow	7
1.7	Scalability Considerations	8
1.8	Application Diagram	9
1.9	Architecture Elements	9
1.10	Design Patterns	9
1.11	Prototype	9

1 Introduction

1.1 Application Description

BalanceBuddy is an application designed to help users improve their lifestyle by monitoring their daily habits. The app tracks various aspects of a person's daily routine such as steps taken, water intake, coffee consumption and food habits. The user will be able to input a goal in the beginning of a day regarding any desired field. At the end of each day, a report is sent to the user, summarizing their activities, progress towards the goal and some recommendations for improvement. BalanceBuddy aims to encourage users change their lifestyle into a healthier and more balanced one.

1.2 Used Technologies

The technologies used for this application are:

- Frontend: React
- Backend: JavaSpring
- Database: MySQL
- API Testing: Postman
- Authentication: JWT (JSON Web Tokens)
- Version Control: Git

I selected these technologies because they make up a robust and efficient technology stack that ensures scalability, flexibility and security for the application. React is chosen for the frontend due to its component-based architecture, which facilitates development. JavaSpring is employed for the backend because of its powerful features for building scalable and reliable applications. MySQL is used as the database for its performance and ease of integration with JavaSpring. Postman is an excellent tool for API testing, offering a user-friendly interface and comprehensive testing capabilities. JWT is utilized for providing a secure and stateless method of verifying users. Finally, Git is selected for version control and dynamic development due to its powerful branching and merging capabilities.

1.3 Functionalities

The main functionalities of the application are:

- **Authentication:**
This functionality allows users to create a new account, to log in or to log out from an already existing account. Users can sign up by providing their email and password. They can log in using their credentials and log out when finished. Sessions are managed using JWT tokens.
- **Habit Tracking:**
This functionality enables users to track their daily habits. The app collects the data regarding steps, which are automatically recorded using a sensor, water and food intake, coffee consumption and others by manual entry.
- **Goal Setting:**
Users can set and manage their personal goals for various habits. They can define targets for each habit and update them later on. A notification system will remind users to work towards their goals.
- **Daily Reports:**
A detailed summary is generated at the end of each day, including the user activities throughout the day and their progress. A recommendation is offered based on the user's performance alongside an encouraging quote.
- **User History:**
A dashboard that provides a comprehensive overview of the user's progress with visual representations such as graphs and charts. These represent the progress and the user is able to view their past activities and habits.

- **Notifications:**

Notifications are sent from time to time to keep users engaged and to remind them to stay on track with their goals. An extra notification is sent at the end of the day to inform the user when their daily report is ready.

1.4 User Stories

1. USER STORY #1 - User Registration

As a new user,

I want to create an account,

So that I can start using the application.

Acceptance Criteria:

AC#1 - Link to Register

Given I am a new user,

When I go to the Login page,

Then I should see a hyperlink to Register.

AC#2 - Register Form

Given I am a new user,

When I follow the hyperlink to Register,

Then I should see a registration form with the following fields:

FieldName	FieldType	Mandatory/Optional Field
Firstname	Free text, [6, 256] characters	Mandatory
Lastname	Free text, [6, 256] characters	Mandatory
Email	Free text, max 256 chars, email format	Mandatory
Password	Free text, min 6 chars, password format	Mandatory
Confirm Password	Should be the same with "Password" field	Mandatory

AC#3 - New Account Creation

Given I am a new user,

When I fill out the registration form correctly and click Register,

Then I should see a "Registration Successful" message, and my account should be created.

Possible errors:

- If the required fields do not meet the validation criteria, an error message is displayed on the screen informing the user what should be changed.
- If some missing fields are detected, another error message is displayed, informing the user.
- If the passwords do not match, a new error message is shown to the user.

2. USER STORY #2 - User Login/Logout

As a registered user,

I want to log in and out from the application,

So that my data is protected.

Acceptance Criteria:

AC#1 - Login Form

Given I am a registered user,

When I go to the Login page,

Then I should see a login form with fields for Username and Password.

AC#2 - Successful Login

Given I am a registered user,

When I enter valid credentials and click Login,

Then I should be redirected to the Main page of the application.

AC#3 - Logout

Given I am a logged-in user,

When I click on the Logout button,

Then I should be logged out and redirected to the Login page.

3. USER STORY #3 - Daily Steps Tracking

As a user,
I want to track my daily steps,
So that I can monitor my physical activity.

Possible errors:

- If some missing fields are detected, another error message is displayed, informing the user.
- If the credentials are invalid, an error message appears informing the user about the mistake.

Acceptance Criteria:

AC#1 - Steps Input

Given I am a logged-in user,
When I go to the Tracking steps page,
Then I should see a field displaying the total number of steps I made today.

4. USER STORY #4 - Water Intake Logging

As a user,
I want to log my water intake,
So that I stay hydrated.

Acceptance Criteria:

AC#1 - Water Intake Input

Given I am a logged-in user,
When I go to the Water Intake page,
Then I should see an input field to enter my daily water intake in liters or glasses or other measuring technique.

AC#2 - Water Intake Logging

Given I am a logged-in user,
When I enter my water intake and click Save,
Then my water intake should be logged and displayed in my activity history.

Possible errors:

- Input is a non-numeric value, so an error is thrown.

5. USER STORY #5 - Goal Setting

As a user,
I want to set daily, weekly or monthly, goals
So that I can improve my lifestyle.

Acceptance Criteria:

AC#1 - Goal Setting Form

Given I am a logged-in user,
When I go to the Goal Setting page,
Then I should see a form to set goals for different parameters.

AC#2 - Save Goals

Given I am a logged-in user,
When I set my goals and click Save,
Then my goals should be saved and displayed on my Dashboard.

AC#3 - Edit Goals

Given I am a logged-in user,
When I edit my goals and click Save,
Then my goals should be saved and displayed on my Dashboard.

Possible errors:

- Input is a non-numeric value, so an error is thrown.

6. USER STORY #6 - Daily Report

As a user,
I want to receive a daily report of my activity,
So that I can analyze my progress.

Acceptance Criteria:

AC#1 - Daily Report View

Given I am a logged-in user,
When I go to the Daily Report page,
Then I should see a summary of my activities throughout the day.

7. USER STORY #7 - User History

As a user,
I want to see a overview of my progress,
So that I can easily track my habits.

Acceptance Criteria:

AC#1 - History View

Given I am a logged-in user,
When I go to the History page,
Then I should see a detailed history of my activity including graphical representations like graphs and charts.

8. USER STORY #8 - Notifications

As a user,
I want to receive timely notifications,
So that I do not forget to input parameters or see the daily report.

Acceptance Criteria:

AC#1 - Notification Settings

Given I am a logged-in user,
When I go to the Notification Settings page,
Then I should see options to set reminders for receiving notifications.

AC#2 - Notification Alerts

Given I am a logged-in user,
When it is time to log steps or water intake,
Then I should receive a notification reminding me to do so.

1.5 Interfaces, Module Communication and Protocol

- **Login/Register:** Here the users can sign up or log in filling up the required fields.
- **User History:** Displays the user's progress with charts and graphs
- **Goal Setting:** Users can set and edit their daily, weekly or monthly goals.
- **Daily Report Page:** Shows a summary of the user's activities during the day and progress towards their goals.
- **Tracking Page:** Allows user to log in their tracking parameters.

The user can freely navigate between the pages as long as it is logged in. The Frontend-Backend communication is handled via RESTful APIs over HTTP. The frontend sends requests to the backend, which processes the data and returns responses. For the authentication, JWT tokens are used to manage

user sessions and secure API endpoints.

The protocol used for this application is HTTP for the connection between server and client and for sending requests, on port 80.

1.6 Message Flow

1. User log in:

The user enters their username and password, the server sends a request of type POST /api/auth/login followed by a body written in a JSON format, with the username and password given by the user, returning a JWT token:

```
{
  "username": "user@example.com",
  "password": "password123"
}
```

2. User registration:

The user enters their information, including, firstname, lastname, username (email), password and confirmed password, the server sends a request of type POST /api/auth/register followed by a body written in a JSON format, with the credentials given by the user, returning the id of the newly created account:

```
{
  "firstname": "John",
  "lastname": "Doe",
  "username": "newuser@example.com",
  "password": "password123",
  "confirmPassword": "password123"
}
```

3. User enters homepage:

When the user wants to return to the homepage of the app, a request of type GET /homepage is sent to the server, returning the "homepage.html" to the user, with the interface of the main page of the application.

4. User views daily report:

If a user selects to view their daily report, a request of type GET /dailyReport/{user_id} will return the daily report page "dailyReport.html".

5. Add a habit:

When a user decides to input information regarding a habit, a request of type POST /habit/add is made, returning a value confirming that the habit addition was made successfully and updates the user's progress.

```
{
  "userId": "12345",
  "habitType": "waterIntake",
  "value": 250,
  "date": "07-08-2024"
}
```

6. Log Sleep:

A request of type POST /sleep/log is made in order to input the parameters regarding the sleeping schedule of the user. As a response, the progress of the user updates and a confirmation message is sent.

```
{
  "userId": "12345",
  "habitType": "waterIntake",
  "value": 250,
  "date": "07-08-2024"
}
```

7. Tracking Exercise:

When the user wants to log details about their physical exercise, the type of the request is going to be POST /exercise/log, where the user can choose the type of exercise he did during the day, and for how much time. In response, a confirmation message is sent and the user's progress updates.

```
{
  "userId": "67890",
  "exerciseType": "running",
  "duration": 30,
  "date": "07-08-2024"
}
```

8. Set a goal:

If a user wants to set a daily goal for a specific habit, a request of type POST /goal/set is made, returning as response a value confirming that the goal setting was made.

```
{
  "userId": "12345",
  "habitType": "steps",
  "targetValue": 10000,
  "period": "daily"
}
```

In order to edit a goal, a PUT /goal/edit request is made, and for deletion of a goal, a DELETE /goal/delete request is made.

9. Generate a Daily Report:

The server makes a request of type POST /report/generate, receiving in return the generated daily report:

```
{
  "userId": "12345",
  "date": "07-08-2024"
}
```

10. Send Reminder Notification:

A request of type POST /notification/reminder is sent containing the user id and the type of announce the notification is for. In response, a value is being sent in order to verify if the notification has been sent correctly or not.

```
{
  "userId": "12345",
  "type": "waterIntakeReminder"
}
```

1.7 Scalability Considerations

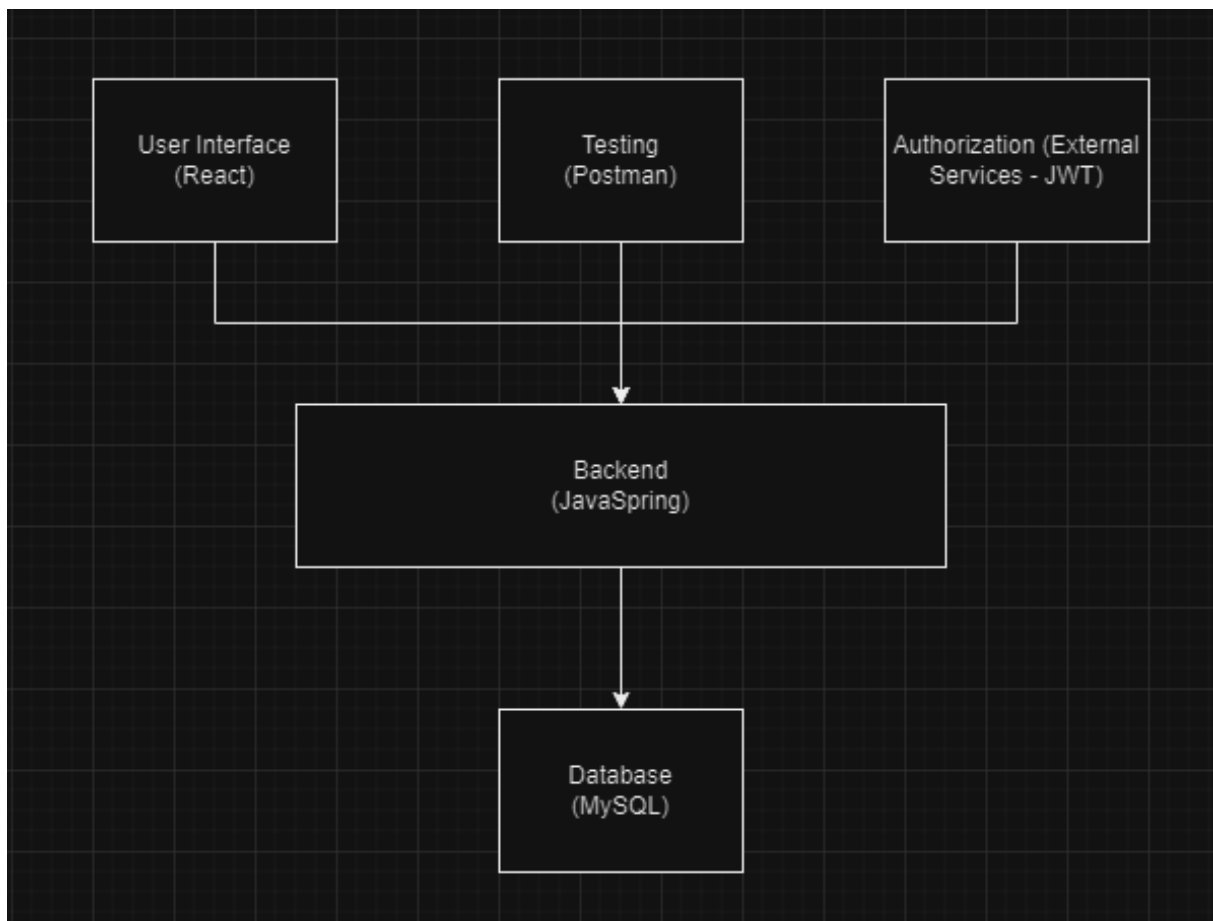
Server Specifications:

- CPU: 6 cores
- Logical processors: 12
- RAM: 32GB
- Storage: 250GB SSD

Load Capacity:

- Concurrent Users: up to 5000
- Requests per Second: up to 500

1.8 Application Diagram



The User Interface (React) provides an interface for BalanceBuddy. The user interacts with the application through the UIT, which sends request to the backend API.

External Services are used for functionalities like sending notifications or handling authentication.

Postman is used for testing the API endpoints exposed by the backend. It simulates requests and checks responses to ensure the API is functioning correctly.

The Backend (JavaSpring) handles all the business logic, exposing API endpoints that the frontend can call to perform operations such as logging habits, setting goals, generating reports, and so on. It interacts with the MySQL database to store and to retrieve data.

The Database (MySQL) stores all the persistent data for the application, including user information, habits, goals, reports.

1.9 Architecture Elements

1.10 Design Patterns

1.11 Prototype