

Angular Certification Training

Certification Project Statement: 2

edureka!

edureka!

NOTE:

There are “**TWO**” certification projects mentioned on the LMS.

You can either “**choose**” certification project-1 or certification project-2.

Certification Project – 2

Problem Statement: Build an online food booking web application using Angular

Note:

- You are provided with a legacy application code on the LMS
- The legacy application is built using PHP/Laravel framework
- You can refer to this application to build the same application using Angular 8 framework

1. Background

FoodOrb is an online food ordering application that allows a user to order food from the nearby food outlet and get it delivered to his/her home.

This web application is a regular stop for millions of users. It has de-facto functions such as user registration, login, reset password, etc. along with that, it will have the core functionality of listing food items and the process to order the same.

The application is built using php and laravel framework, and had the below-listed disadvantages:

1. The complexity to code is high, and large changes are required for updates, i.e., no incremental updates
2. Tight coupling between view and controller, i.e., lack of flexibility for modification of either backend or frontend alone
3. Inefficient data access in view
4. Decreases development speed as the application gets bigger
5. Redundant implementation of Business Logic in controller and View

All these challenges led to unsatisfied user experience. So, the company decided to upgrade the website using the Angular Framework to focus on building a responsive and customer-oriented single page web application.

2. Goal

To migrate the legacy application to a RESTful web-application built using the Angular 8 framework.

3. Use Cases:

The application is designed to provide user-specific functionality. We will have three users for this application:

- **Non-Registered** - Users who are not registered
- **Registered** - Users who have registered & login to the website
- **Admin** - Users who have administrative capabilities

4. Web Application Requirement:

The web application will be developed using the latest frontend technologies: HTML, CSS, JavaScript, and Angular 8.

You will be provided backend API endpoints developed in laravel 7 framework using PHP language.

5. Web Application Implementation:

Frontend

The frontend of the application will be developed using Angular 8. The frontend will be a single page application, which will be developed in modules that represent roughly the feature of the website.

Modules:

1. **Registration** - This consists of all user-related pages such as registration page, login page, forgot password page, profile page, and settings page
2. **Listing** - This module consists of the food listings and feed pages where the user can browse and view the details of the food items
3. **Ordering** - This module consists of ordering and tracking pages, where users can place orders and track the placed order
4. **Friends List** - This module consists of browsing and connecting friends and displaying friend lists
5. **Services** - This module consists of talking to the backend API's and displaying data on the UI

Backend

The backed API is developed in laravel 7 a framework developed using PHP language.

There are a set of API endpoints that are mentioned below that will be used by the frontend application. The details of the API endpoints are as below.

base_url = http://localhost/ or http://<domain or ip address>/

1. Registration:
 - <base_url> + api/v1/register
2. Login:
 - <base_url> + api/v1/login
3. Forgot Password:
 - <base_url> + api/v1/forgotpassword
4. Users:
 - a. List Users: <base_url> + api/v1/users
 - b. User Details: <base_url> + api/v1/users/{user_id}
5. Settings:
 - <base_url> + api/v1/settings
6. Feed (Food Listing):
 - <base_url> + api/v1/feed
7. Orders:
 - <base_url> + api/v1/orders
8. Order:
 - <base_url> + api/v1/orders/{order_id}
9. Order Status:
 - <base_url> + api/v1/orders/<order_id>/status
10. Friends:
 - <base_url> + api/v1/friends

6. Testing:

Testing for frontend will be done using jasmine & karma. Example unit tests will be written in order to convey the testing methodology used for testing of modular and component-based single-page applications built using angular.

7. Feature Flow:

The application will have around 9 to 10 flows for various features described in the previous sections of this document. The details of each flow are described below and raw wireframes and provided for a reference implementation.

1. Registration Flow
2. Login Flow
3. Forgot Password Flow
4. Home Page Flow
5. Network Page Flow
6. Settings Page Flow
7. Friend-List Page Flow
8. Order Flow
9. Tracking Flow

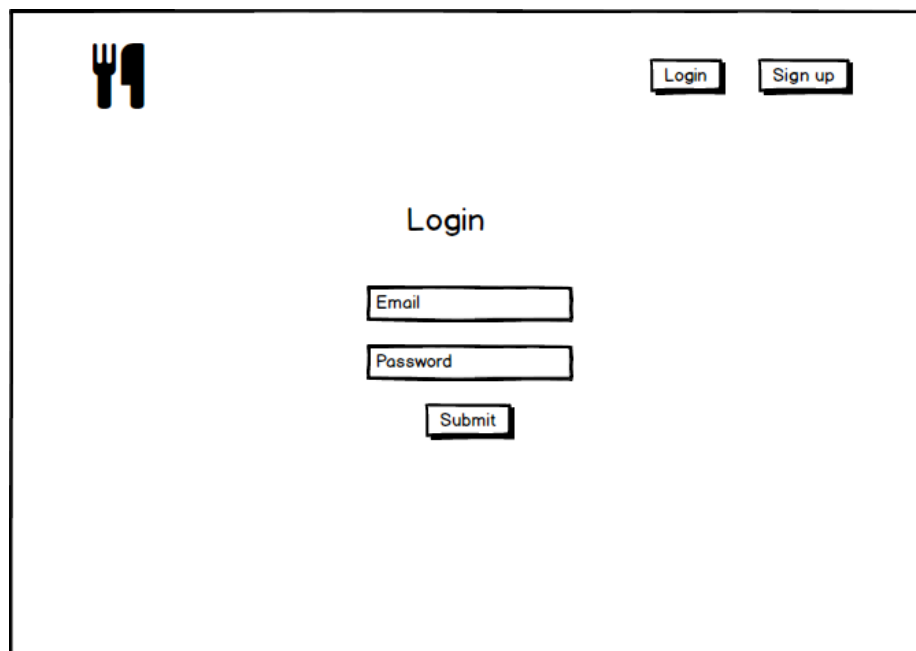
1. Registration Flow



Wireframe of the Registration Flow page:

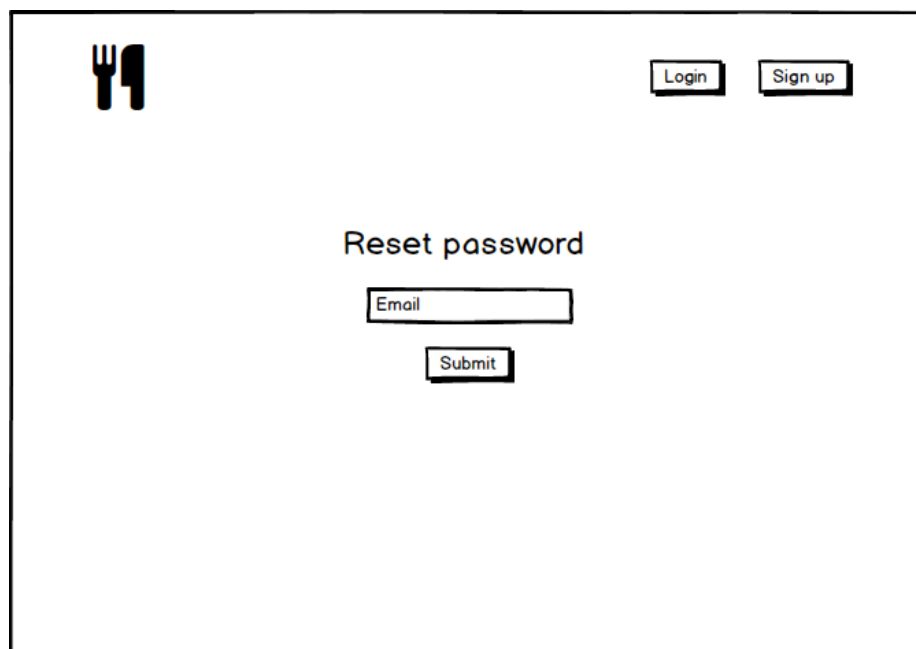
- Header: Fork and knife icon on the left; Login and Sign up buttons on the right.
- Main Content: Sign up! title.
- Form Fields: Phone number, Name, Email, Password.
- Submit Button: Submit.

2. Login Flow



The mockup shows a login interface. In the top-left corner is a logo consisting of a fork and a knife. In the top-right corner are two buttons: "Login" and "Sign up". Centered on the page is the heading "Login". Below the heading are two input fields: "Email" and "Password". Below the "Password" field is a "Submit" button.

3. Forgot Password Flow

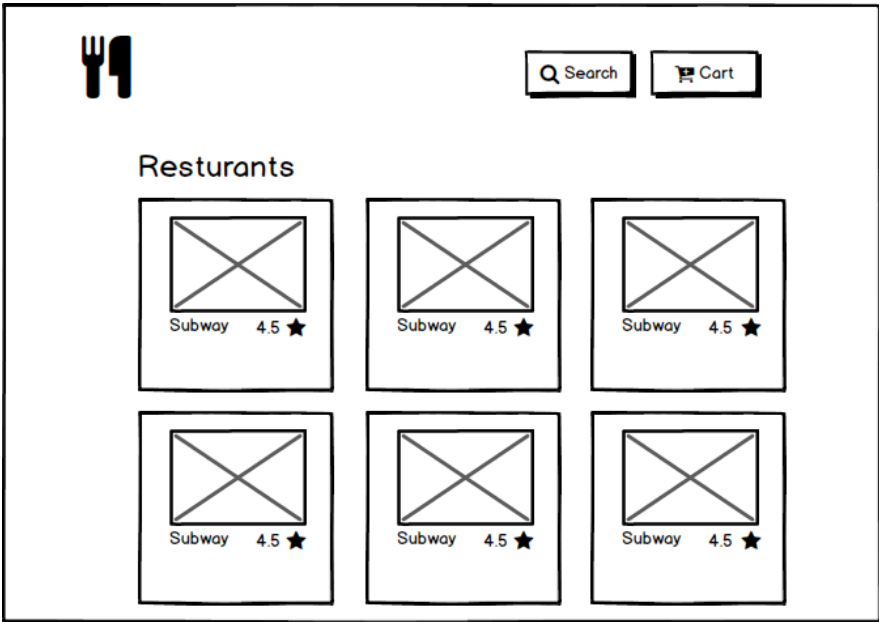


The mockup shows a "Reset password" interface. In the top-left corner is a logo consisting of a fork and a knife. In the top-right corner are two buttons: "Login" and "Sign up". Centered on the page is the heading "Reset password". Below the heading is an "Email" input field. Below the "Email" field is a "Submit" button.

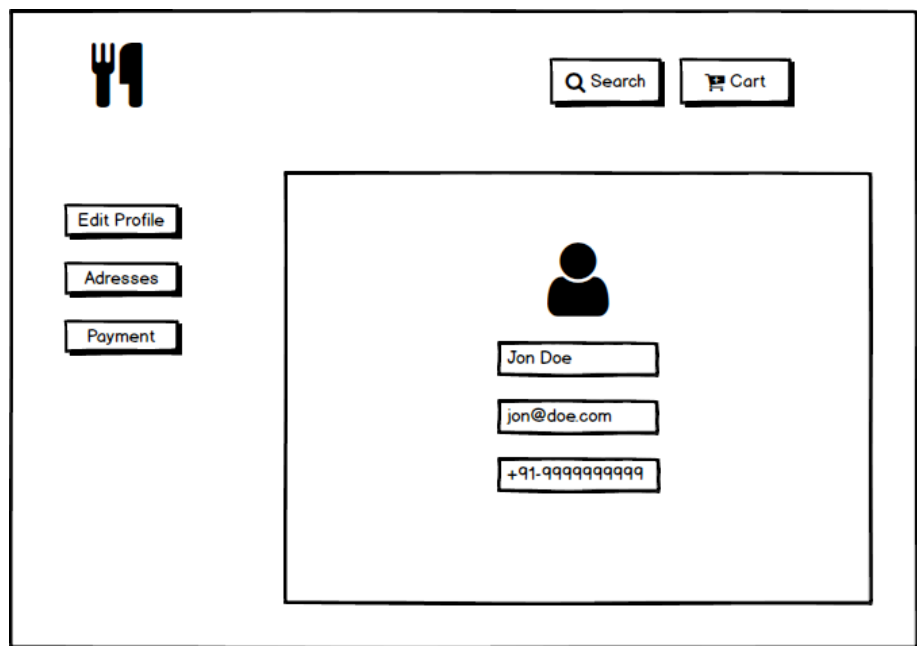
4. Home Page Flow



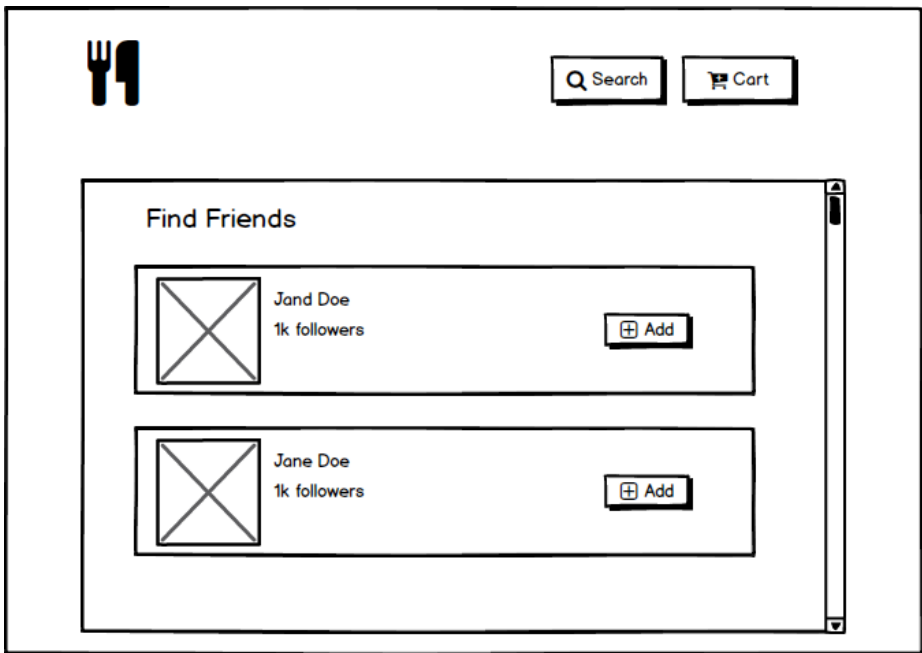
5. Network Page Flow



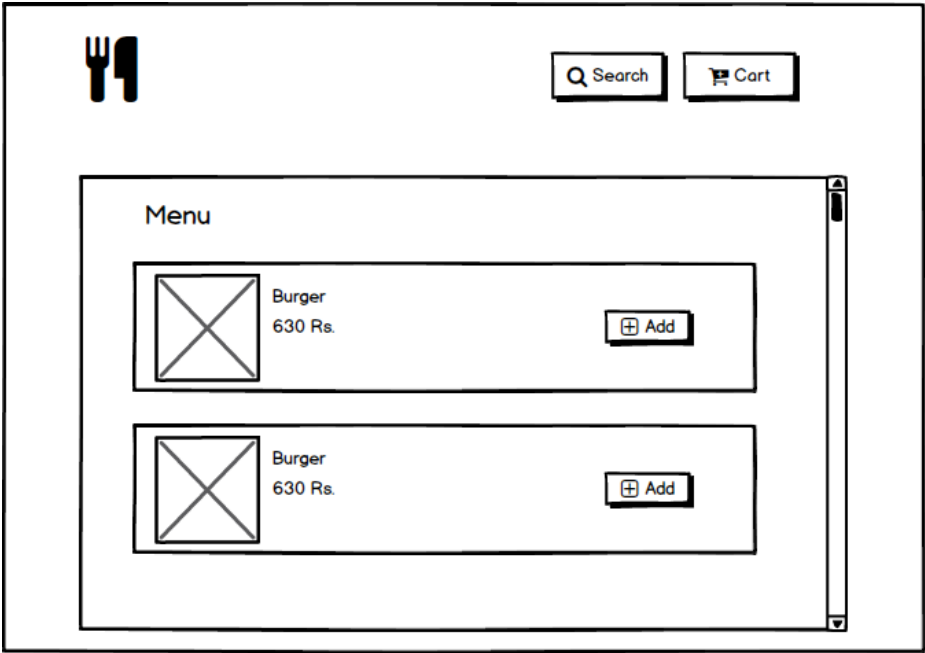
6. Settings Page Flow



7. Friend-List Page Flow



8. Order Flow



9. Tracking Flow

