Angular: Platform and Framework for Web Application Development

Luis Miguel Garcia Marin
Fulda University of Applied Science
luis-miguel.garcia-marin@informatik.hs-fulda.de

Contents

ABSTRACT

In this paper we present an introduction to Angular [1], a framework and platform that enables the development of client-side single-page applications (SPA) through the use of TypeScript and HTML. Additionally, we will provide a practical example of developing a web application using Angular, which allows users to share cars and travel together. The platform enables users to publish travel routes and fares, while others can join in. The implementation of the application is based on dummy data and is only available on the front-end. The code repository is provided in the references [2].

1. INTRODUCTION

Car sharing has become increasingly popular in recent years. This approach allows users to reduce costs and improve travel efficiency. However, coordination between users can often be complicated. To address this problem, a web application has been developed in Angular that allows users to share cars and travel together efficiently.

1.1 What is Angular?

Angular [1] is a platform and framework for building single-page client applications (SPA) using HTML and TypeScript. It is a set of tools and libraries that provide a structured and consistent approach to web application development.

It was developed by Google and released in September 2016, and has continued to be supported ever since. However, before its public release, Angular was used exclusively by Google (first version was released internally in 2009 [3]), but in 2016 they decided to make it public. Since then, it has become a popular tool for developers around the world due to its ability to create dynamic and scalable web applications.

1.2 Features of Angular

- HTML and TypeScript: Angular is based on HTML and TypeScript, a language written on top of JavaScript that provides types, improved error checking and code maintainability.
- Components [4]: Angular provides a powerful component based architecture that allows develop complex applications by composing simple components. This makes it easy to reuse and share code, and simplifies the development of large-scale applications.
- Dependency Injection: Angular uses dependency injection to manage object creation and simplify the development of complex applications.
- Declarative Templates: Angular utilizes declarative templates to separate application logic from presentation, making it easier to maintain and scale applications.
- Routing [5]: Angular includes a routing system that allows us to navigate between the different views of our application in a coherent and structured way.
- Progressive Web Apps (PWA): Angular provides support for building Progressive Web Apps, which allows the user to install the application and use it in

offline mode as if it were a native application. This is very useful to improve the user experience and portability.

1.3 How to create a project

In order to create a project in Angular [4] you need to follow these steps:

1. Install a code editor. To edit the project, you need to have a code editor installed on your system. Visual Studio Code is a good option as it has extensions to facilitate development in Angular.
2. Install Node.js and npm (Node Package Manager), which are required by Angular.
3. Install the Angular client. To install the Angular client, run this command in a command line tool: `npm install -g @angular/cli`. It will install the Angular client globally on your system.
4. Create a new project. To create a new project, execute: `npm new [project-name]`. This command creates a new folder with the name of your project and generates the basic structure of the project in it.
5. Run the project. Navigate to the project folder using the command `cd [project-name]` and then execute the following command: `ng serve -o`. This command will compile the project and start it in our default browser, allowing us to see the changes in real time as we make them.

2. STATE OF THE ART

2.1 Angular vs React

In the field of frameworks used for building web applications, we can see that two of the most popular frameworks are React and Angular. Here are some of the main differences between the two:

- Flexibility: React is known for its flexibility, giving developers more freedom to structure their projects and choose their preferred syntax. Angular, on the other hand, offers a stricter project structure and syntax that can be beneficial for novice developers or large teams needing to follow a defined standard.
- Origin: Google develops and maintains Angular, while Facebook is responsible for React.
- Typing: Angular is based on TypeScript. React, on the other hand, only requires JavaScript and JSX for component creation.
- Templates: Angular uses HTML templates for component creation, which can be beneficial for web designers and make it easier to create reusable components. In contrast, React uses JSX, a JavaScript extension that allows for component creation with a syntax similar to HTML.
- Data binding: Angular allows for bidirectional data binding, where changes to the data model are automatically reflected in the view. React, however, uses unidirectional data binding, requiring changes in the data model to be explicitly passed to the view using props [6].

2.2 Angular information sources

Since Angular has a very big community, there are several sources where you can learn more about this framework.

YouTube is a great resource for finding channels that offer tutorials for beginners. For example, 'Amigos Code' [7] provides straightforward tutorials that are easy to follow. 'FreeCodeCamp.org' [8] and 'Traversy Media' [9] are also excellent channels to check out. Additionally, always is a good option to take courses in platforms like Udemy [10], Coursera or others similar.

Moreover, there are several resources available to developers looking for help with Angular. The official Angular documentation [4] is a recommended source, as it provides comprehensive information about the framework. In addition, there are several forums such as 'Stack Overflow' and 'Reddit', where developers can ask specific questions and get answers from the community. These resources can be useful for troubleshooting, gathering information and learning best practices related to Angular development.

3. METHODOLOGY

To illustrate how Angular is used in web application development, a practical example of an application that allows users to carpool and travel together is presented [2]. The app provides a platform for users to post routes and prices and for other users to decide if they want to join. The application is based on dummy data and only has a front-end implementation.

In the following paragraphs, we are going to see different parts of the written code (components, models, services...) in Angular for the development of this application.

3.1 App Module and Routes

One of the important parts of the code is the app.module.ts file. It is the root module of an Angular application, and it sets up the dependency injection system, which is used to provide instances of services and other dependencies to components and other parts of the application.

In the Figure 1, the appRoutes array defines the routes for the application and maps each path to a corresponding component. When a user navigates to a path in the application, Angular uses the defined routes to determine which component to render. For example, the IndexPageComponent is associated with the root path (''), while the LoginPageComponent is associated with the 'login' path.

The @NgModule decorator is used to define the metadata for the root module of the application, which includes the declarations of components, imports of other external modules, and providers properties for application services.

```
const appRoutes: Routes=[
  {path:'', component: IndexPageComponent},
  {path:'login', component: LoginPageComponent},
  {path:'register', component: RegisterPageComponent},
  {path:'newtrip', component: NewTripPageComponent},
  {path:'trips', component: TripsPageComponent},
  {path:'mytrips', component: MyTripsPageComponent}
]

@NgModule({
  declarations: [
    AppComponent,
    IndexPageComponent,
    NavarComponent,
    FooterComponent,
    LoginPageComponent,
    RegisterPageComponent,
    NewTripPageComponent,
    TripsPageComponent,
    TripCardComponent,
    MyTripsPageComponent
  ],
  imports: [
    BrowserModule,
    AppRoutingModule,
    FormsModule,
    RouterModule.forRoot(appRoutes),
    HttpClientModule,
    GoogleMapsModule
  ],
  providers: [UsersServiceService,
  TripServiceService],
  bootstrap: [AppComponent]
})
```

Figure 1. Declaration of routes and modules with components and their corresponding routes.

3.2 Models

We use models in Angular to define objects and we use this models in the application. In the Figure 2 we can see how we define a user in Carida. The class has a constructor method to initialize the properties of the objects.

In Carida we also have defined the model Trip, which has the properties: id, start, finish, price, date, driver, freeSeats, occupiedSeats, startLatitude, startLongitude, finishLatitude, and finishLongitude. The definition of this model is pretty similar to the model user.

3.3 Services

In the Figure 3 we can see one of our services. A service is a class that manages all the behavior and business logic related to a model (if we were connected to a back-end API, these services would be in charge of sending and receiving the information through the API).

In this case we can see the class TripServiceService, which is in charge of all the business logic related to the trips, such as adding a new trip (addTrip), find a trip (findTrip) and get all the trips that we have until now (getTrips). The same goes with the class UserServiceService, which is in charge of all the behavior related to the users (registration, login...).

We also have another service called GeocodingService, which we use to work with the GoogleMaps module in order to geographically encode the latitude and longitude of

```
import { Trip } from "./trip.module";

export class User{

  name?:string;
  email?:string;
  password?:string;
  trips?:Trip[];

  constructor(email?:string, password?:string,
    name?:string, trips?:Trip[]){
    this.email = email;
    this.password = password;
    this.name = name;
    this.trips = trips;
  }
}
```

Figure 2. User class created in Angular.

the points that we save (geoCodeLatLng), and also to get the name of a location of a given position with latitude and longitude (getLocation). These functions make an API call to the end points that Google Geocoding offers to work with their maps and Geocoder functionalities. It is important to mention that you need a Google Cloud account to use these tools [11].

```
@Injectable({
  providedIn: 'root'
})
export class TripServiceService {

  trips: Trip[] = [
    ...
  ];

  singleID: number = 5;

  constructor() { }

  addTrip(trip:Trip){
    trip.id = this.singleID;
    this.singleID++;
    this.trips.push(trip);
  }

  findTrip(trip:Trip): Trip | undefined{
    return this.trips.find(TripToFind => TripToFind.start == trip.start
                          && TripToFind.finish == trip.finish);
  }

  getTrips(): Trip[]{
    return this.trips;
  }

}
```

Figure 3. Trip Service class created in Angular.

3.4 App Component

The template code for the root component of the Angular car sharing application looks as follows:

```
<app-navar></app-navar>
<br>
<router-outlet></router-outlet>
<br>
<app-footer></app-footer>
```

This template includes the navigation bar and footer components, and uses the router outlet to render the content of the current page. This allows the application to have a consistent layout and navigation scheme across all pages.

The router-outlet tag is a special directive that tells Angular where to render the content of the current page. This is where the different components of the application, such as IndexPageComponent or NewTripPageComponent, will be displayed when the user navigates to them using the navigation links in the navigation bar [5].

3.5 Trip Page Component

The Trips Page Component is responsible for displaying all available trips to the user, and it is also responsible for filtering out any trips that the user has already booked. Its template code can be seen in Figure 4.

```
<div class="container bg-palid-green
    rounded shadow">
  <p class="text-center fs-1 p-4 fw-bold">
    Trips
  </p>
</div>
<div class="container" id="main-container">
  <ng-container *ngFor="let trip of trips">
    <app-tripCard [trip]="trip"
        [booked]="false">
    </app-tripCard>
  </ng-container>
</div>
```

Figure 4. Trip Page Component template code created in Angular.

The component iterates over the filtered list of trips using the ngFor directive and creates an instance of the tripCard component for each trip. The tripCard component is passed to the trip object as an input parameter, which it uses to display the trip details to the user. The booked attribute is set to false because the component is displaying all available trips, and the user has not yet booked any trips.

4. RESULTS

The application developed using Angular results in a simple and intuitive website that makes car sharing easy for everyone, despite the fact that it has no back-end at the moment. All the data is stored in the application itself, which means that all the changed data during a user session is deleted once the page is reloaded.

4.1 Main Page

In the Figure 5, we can see the main page, with the navbar, that offers the options to Login or Register, and the footer. There is also a brief description of the page, along with two cards that showcase the various roles available to users.

The login and register pages have the typical fields and they are effective. The design of these pages, including the main page, have proven to be simplistic and attractive.

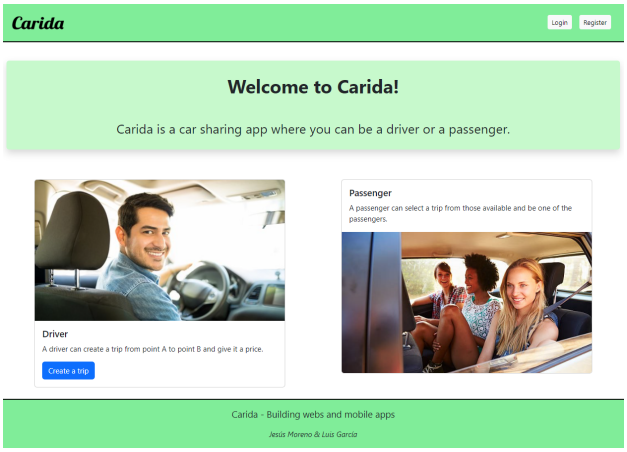

Figure 5. Main page of the car sharing application (Carida).

4.2 Trip Page

We can see that the “Trips” page (showed in Figure 6) is a highly usable and effective feature that provides users with an easy way to browse available trips. The display of routes on map (showing the optimum route calculated by Google Maps API) and other map options were found to be effective in improving user experience and finding relevant trips.

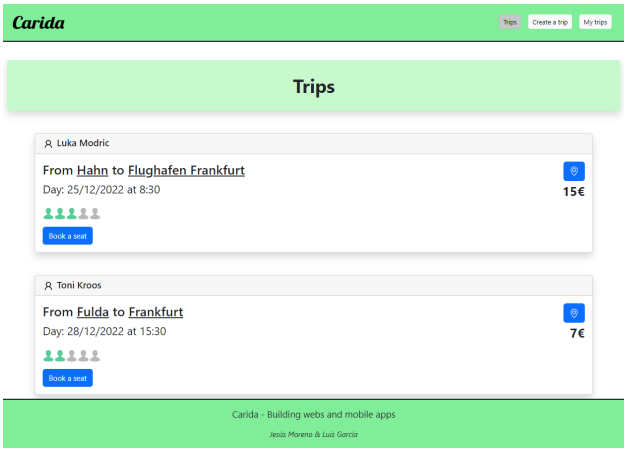

Figure 6. Page of Carida where you can see all the trips available to book.

4.3 Create a trip

The Create Trip page (showed in Figure 7) of the Angular car sharing application enables the users to create a new trip by providing the necessary details (with error validation), including the start and end points, date and time of the trip, the number of available seats, and the price per seat. The page has a user-friendly interface that includes a map window (using Google Maps module) that allows users to select the points for the start and end locations of the trip.

After the user has provided all the necessary details, they can click on the ”Create Trip” button to create the trip. Overall, the Create Trip page of the Angular car sharing

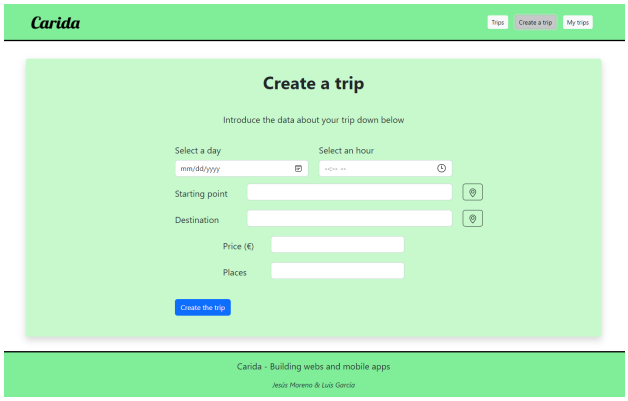

Figure 7. Page for creating a trip to share.

application provides a user-friendly interface that allows users to create new trips easily and efficiently.

4.4 Join a trip

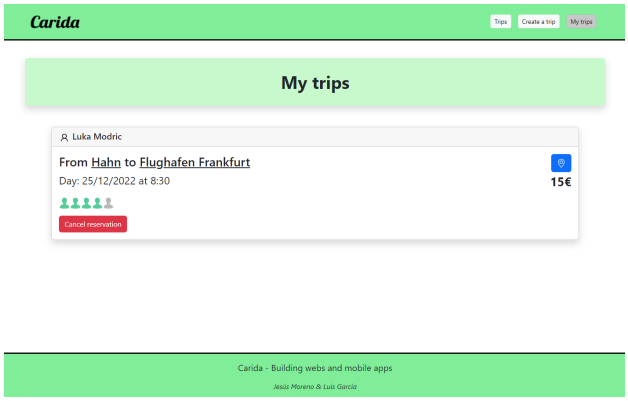

Figure 8. My Trips Page, which displays the trips that you have joined.

We can join a trip by clicking the ”Book a seat” button on the trip displayed in ”Trips” page. After joining, the trip will be displayed on the ”My trips” page (showed in Figure 8), and the user can view the trip details along with the booked seat. If the user decides to cancel their reservation, they can click on the ”Cancel reservation” button, which will remove the trip from their ”My trips” page and free up the seat for others to book. Overall, this feature enhances the user experience and increases the accessibility of carpooling.

5. DISCUSSION

Overall, the application developed in Angular [2] represents a significant advance in car sharing and offers an effective solution for the coordination and sharing of trips. Although it is based on dummy data and only has a front-end implementation, the system provides a simplified and efficient user experience for travel coordination and sharing.

However, there is a lot of room to improve and expand its functionality in the future. For example, the integration of a back-end to allow user authentication and real data management would be a major improvement. In addition, the inclusion of additional features, such as the possibility of making online payments, could also be useful to improve the user experience.

As we have seen, Angular is a very complete and powerful framework for developing web applications. Although we don’t have enough experience with React to do a very comprehensive comparison, we have found working with Angular to be very satisfying. Although its learning curve can be a bit steep at first, once you get used to its structure and wide variety of tools, it can be very efficient and powerful in creating high-quality web projects. Also, it’s important to note that Angular is not the best choice for native mobile app development (Flutter or React Native can be better options for that case), but if you are looking for a solid framework for developing web applications, Angular is an excellent option.

6. REFERENCES

[1] “Angular official site,” https://angular.io/, accessed: 2023-03-29.

[2] L. M. G. Marín and J. M. Durán, “Car Sharing App (BWMA-Carida),” https://github.com/jesu-m0/BWMA-Carida, 2022, [Online; accessed 29-March-2023].

[3] “AngularJS Wikipedia site,” https://en.wikipedia.org/wiki/AngularJS, accessed: 2023-03-29.

[4] “Angular official documentation (Angular Docs),” https://angular.io/docs, accessed: 2023-03-29.

[5] ZeroesAndOnes, “How to create navigation in Angular 9 Application using Angular Router,” https://zeroesandones.medium.com/how-to-create-navigation-in-angular-9-application-using-angular-r 2020, [Online; accessed 29-March-2023].

[6] B. Gauca, “Angular vs. React - A comparison,” http://work.haufegroup.io/Angular-VS-React/, 2017, [Online; accessed 29-March-2023].

[7] “Amigoscode Youtube channel,” https://www.youtube.com/@amigoscode, accessed: 2023-03-29.

[8] “freeCodeCamp.org Youtube channel,” https://www.youtube.com/@freecodecamp, accessed: 2023-03-29.

[9] “Traversy Media Youtube channel,” https://www.youtube.com/@TraversyMedia, accessed: 2023-03-29.

[10] M. Schwarzmüller, “Udemy Course: Angular - The Complete Guide (2023 Edition),” https://www.udemy.com/course/the-complete-guide-to-angular-2/, 2023, [Online; accessed 29-March-2023].

[11] A. Zia, “Using Google Geocoding API in Angular,” https://arminzia.com/blog/using-google-geocoding-api-in-angular/, 2021, [Online; accessed 29-March-2023].