REBOOT SESSION – VUE.JS

VUE.JS

ae architects for business & ict


```vue
<template>
    <div class="quest-list">
        <v-card v-for="(quest,index) in quests" :key="index">
            <v-container>
                <quest-tile v-bind:quest="quest"></quest-tile>
            </v-container>
        </v-card>
    </div>
</template>

<script lang="ts">
import {Component, Vue} from 'vue-property-decorator';
import QuestTile from './QuestTile.vue';
import {questService} from '@/services/Quest.service';
import {IQuest} from '@/models/IQuest';

@Component({ components: { QuestTile } })
export default class QuestList extends Vue {
  private quests: IQuest[] = [];

  private created() {
    this.quests = questService.getAllQuests();
  }
}
</script>

<style lang="scss" scoped>
  .quest-list {
    display: flex;
    flex-direction: column;
    justify-content: space-around;
  }
</style>
```

```vue
<template>
  <div class="about">
    <v-card v-for="(item, index) in items" :key="index">
      {{ item }}
    </v-card>
  </div>
</template>

<script lang="ts">
import {Component, Vue} from 'vue-property-decorator';

@Component({})
export default class About extends Vue {
  private items: string[] = [];

  private created() {
    this.items = ['Bike Quest', 'Car Quest', 'Chair Quest'];
  }
}
</script>
```

Git

```
git clone <repo>
```

```
git fetch
```

```
git checkout <branch>
```

```
git add <change>
```

```
git commit –m “<message>”
```

```
npm install -g @vue/cli
```

```
git clone https://github.com/AE-nv/technovate
cd technovate
```

```
git checkout bike-quest/exercise-1
```

```
cd karma-village
npm i
```

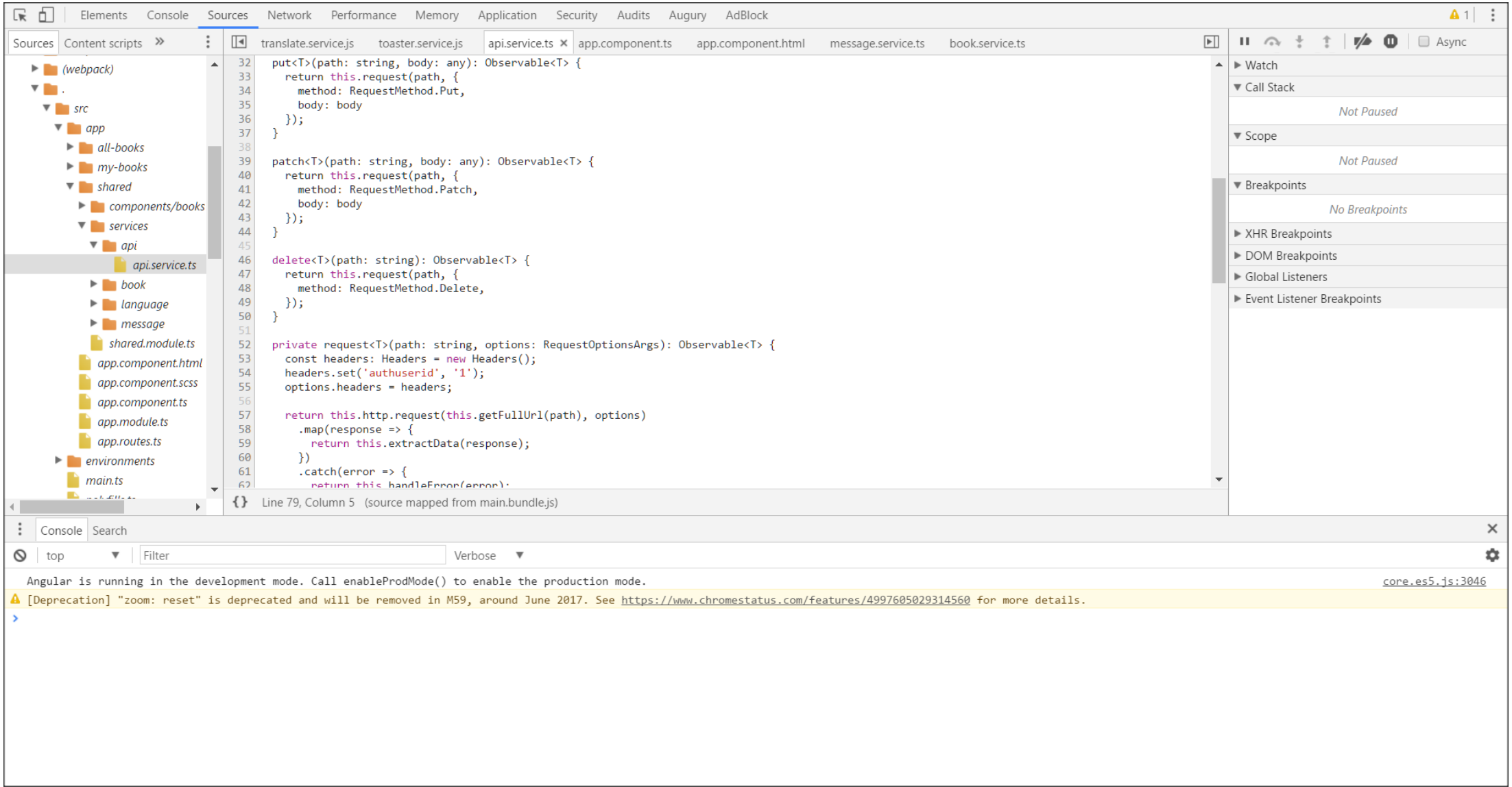
```
npm run serve
```

Go to http://localhost:8080

src/components/quests => Fix the TODO's!

Chrome Dev Tools

https://developer.chrome.com/devtools



Bike Quest – Exercise 1

- Add a v-textarea
- Display input in v-card

BikeQuest.vue

```
<v-card>
    <v-container>
        <v-textarea auto-grow v-model="text" box rows="1" label="Response"></v-textarea>
        <v-card class="response-card">
            {{ text }}
        </v-card>
    </v-container>
</v-card>
```

```
export default class BikeQuest extends Vue {
    public text: string = '';


    get isQuestComplete(): boolean {
        return false;
    }
```

Bike Quest – Exercise 2

- Add a v-btn
- Add input to list when button is clicked
- Display all responses in list

BikeQuest.vue

```
<v-card>
    <v-container>
        <v-card  v-for="(res, index) in responses" :key="index" class="response-card">
            {{ res }}
        </v-card>
        <v-textarea auto-grow v-model="text" box rows="1" label="Response"></v-textarea>
        <v-btn @click="addResponse" :disabled="!text">Submit</v-btn>
    </v-container>
</v-card>
```

BikeQuest.vue

```
public text: string = '';
public responses: string[] = [];

get isQuestComplete(): boolean {
    return false;
}

public addResponse(): void {
    this.responses.push(this.text);
    this.text = '';
}
```

Bike Quest – Exercise 3

- Complete quest when response contains words:
  - pedals
  - saddle
  - wheels

BikeQuest.vue

```
get isQuestComplete(): boolean {
    let completed = false;
    for (const res of this.responses) {
        if (res.indexOf('saddle') > -1 && res.indexOf('pedals') > -1 && res.indexOf('wheels') > -1) {
            completed = true;
        }
    }
    return completed;
}
```

Bike Quest – Exercise 4

- Move text-area to TextAreaBox.vue
- Emit input to parent
- Add text-area-box to BikeQuest.vue

TextAreaBox.vue

```
<template>
    <div class="text-area-box">
        <v-textarea auto-grow v-model="text" box rows="1" :label="label"></v-textarea>
        <v-btn @click="addResponse" :disabled="!text">{{ btnText }}</v-btn>
    </div>
</template>
```

```vue
<script lang="ts">
    import { Component, Prop, Vue } from 'vue-property-decorator';

    @Component({})
    export default class TextAreaBox extends Vue {
        @Prop({default: 'Submit'})
        public btnText: string;

        @Prop({default: 'Response'})
        public label: string;

        private text: string = '';

        private addResponse() {
            this.$emit('text', this.text);
            this.text = '';
        }
    }
</script>
```

```vue
<template>
    <div class="text-area-box">
        <v-textarea auto-grow v-model="text" box rows="1" :label="label"></v-textarea>
        <v-btn @click="addResponse" :disabled="!text">{{ btnText }}</v-btn>
    </div>
</template>
```

BikeQuest.vue

```
<v-card>
    <v-container>
        <v-card  v-for="(res, index) in responses" :key="index" class="response-card">
            {{ res }}
        </v-card>
        <text-area-box @text="addResponse"></text-area-box>
    </v-container>
</v-card>
```

```
import TextAreaBox from '@/components/Shared/TextAreaBox.vue';

@Component({
    components: { NavigationComponent, TextAreaBox },
})
export default class BikeQuest extends Vue {
```

```
public addResponse(value: string): void {
    this.responses.push(value);
}
```

Car Quest – Exercise 1

- Add text-box-area to CarQuest.vue
- Call the CarSearchService when button is clicked
- Display the result

CarQuest.vue

```
<v-card>
    <v-container>
        <h3 class="quest-header">
            Search for Cars
        </h3>
        <text-area-box @text="searchForCars" :btnText="'Search Cars'"
                :label="'Car keywords go here'">
        </text-area-box>
        <div v-for="(car, index) in foundCars" :key="index">
            <v-card class="car-card">
                <div>
                    {{car.make_display}} - {{car.model_trim}}
                </div>
            </v-card>
        </div>
    </v-container>
</v-card>
```

CarQuest.vue

```
export default class CarQuest extends Vue {
    public foundCars: Car[] = [];

    get isQuestComplete(): boolean {
        return false;
    }

    public searchForCars(value: string): void {
        this.foundCars = carSearchService.search(value);
    }

}
```

Car Quest – Exercise 2

- Implement the search function in the CarSearchService

CarSearchService.vue

```
public search(keyword: string): Car[] {
    const filter = this.carApiResponse.Trims.filter((car: Car) =>
            car.model_trim.toLowerCase().indexOf(keyword.toLowerCase()) > -1 ||
            car.make_display.toLowerCase().indexOf(keyword.toLowerCase()) > -1);
    return filter;
}
```

Car Quest – Exercise 3

- Implement functions to recommend and unrecommend cars
- Complete quest when recommended cars contains BMW

CarQuest.vue

```
<div v-for="(car, index) in foundCars" :key="index">
    <v-card class="car-card">
        <div>
            {{car.make_display}} - {{car.model_trim}}
        </div>
        <div>
            <v-btn flat icon @click="recommend(car)">
                <v-icon>thumb_up</v-icon>
            </v-btn>
        </div>
    </v-card>
</div>
```

CarQuest.vue

```
public recommend(value: Car): void {
    this.recommendations.push(value);
    this.foundCars.splice(this.foundCars.indexOf(value), 1);
}

public unrecommend(value: Car): void {
    this.foundCars.push(value);
    this.recommendations.splice(this.recommendations.indexOf(value), 1);
}
```

```
get isQuestComplete(): boolean {
    return !!this.recommendations.find((car: Car) => car.make_display.toLowerCase().indexOf('bmw') > -1);
}
```

Car Quest – Exercise 4

- Replace recommended and result list by CarList component

CarList.vue

```
<template>
    <div class="car-list">
        <div v-if="cars.length === 0">
            No cars found...
        </div>

        <div v-for="(car, index) in cars" :key="index">
            <v-card class="car-card">
                <div>
                    {{car.make_display}} - {{car.model_trim}}
                </div>
                <div>
                    <v-btn flat icon @click="carClicked(car)">
                        <v-icon>{{btnText}}</v-icon>
                    </v-btn>
                </div>
            </v-card>
        </div>
    </div>
</template>
```

CarList.vue

```
import { Component, Prop, Vue } from 'vue-property-decorator';
import { Car } from '../../services/CarSearchService';

@Component({})
export default class CarList extends Vue {
    @Prop({default: 'Submit'})
    public btnText: string;

    @Prop()
    public cars: Car[];

    private carClicked(value: Car) {
        this.$emit('carClicked', value);
    }
}
```

Chair Quest – Exercise

- Add a DropZone component
- Show list of files that are uploaded
- Call the GoogleVisionApi to determine characteristics
- Provide user feedback about loading and result API-call

ae®

architects
for business
& ict

ae.be

Confidential and Ownership Clause

Contact: marketing@ae.be