МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Івано-Франківський національний технічний університет нафти і газу

Кафедра інформаційно-телекомунікаційних технологій та систем

Л. О. Штаєр

ТЕХНОЛОГІЇ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

ЛАБОРАТОРНИЙ ПРАКТИКУМ
ЧАСТИНА 2

Івано-Франківськ
2017

УДК 004.42
ББК 32.973-01
Ш-87

Рецензент:
Незамай Б. С. кандидат технічних наук, доцент кафедри математичних методів у інженерії Івано-Франківського національного технічного університету нафти і газу

Рекомендовано методичною радою університету (протокол № __ від __.__.2017 р.)

Штаєр Л. О.

Ш-87 Технології розробки програмного забезпечення: лабораторний практикум. – Ч. 2. – Івано-Франківськ: ІФНТУНГ, 2017. – 64 с.

МВ 02070855-____-2017

Лабораторний практикум містить методичні матеріали для проведення лабораторних занять з дисципліни “Технології розробки програмного забезпечення” у другому семестрі вивчення даної дисципліни. Розроблений відповідно до робочої програми навчальної дисципліни “Технології розробки програмного забезпечення”. Може бути використаний студентами денної та заочної форм навчання.

Призначений для підготовки фахівців за напрямом підготовки “Системна інженерія”.

УДК 004.42
ББК 32.973-01
МВ 02070855-____-2017

ЗМІСТ

ЗАГАЛЬНІ МЕТОДИЧНІ ВКАЗІВКИ

Дисципліна «Технології розробки програмного забезпечення» є профілюючою, забезпечує теоретичну та інженерну підготовку фахівців з напряму «Системна інженерія».

Мета викладання дисципліни "Технології розробки програмного забезпечення " — одержання студентами теоретичних основ з моделювання, розробки та проектування на основі шаблонів та каркасів програмного забезпечення, а також вивчення основних стандартів, вимог та готових конструкцій, які використовуються при розробці та проектуванні програмних додатків, студенти набувають практичних навичок проектування програмних продуктів з використанням сучасних методів та засобів.

При вивченні курсу від студентів вимагається володіння навичками та методами, одержаними при вивченні курсів "Програмування", "Комп’ютерні мережі".

В результаті вивчення дисципліни:

- студент повинен знати: моделі життєвого циклу програмного забезпечення (ПЗ), процеси та етапи розроблення ПЗ, методології проектування, сучасні технології проектування ПЗ, їх характеристики та особливості, а також застосування їх на практиці; мету та задачі проектування програмних засобів; мову моделювання UML; реалізацію клієнтської частини web-орієнтованих програмних продуктів; шаблон проектування MVC.
- студент повинен вміти: користуватись методами об'єктно-орієнтованого проектування програмного забезпечення та застосовувати ці методи на практиці; використовувати мову моделювання UML для проектування програмного забезпечення; користуватися шаблонами проектування та розуміти їх роль в програмуванні, реалізовувати додатки на AngularJS.

Друга частина лабораторного практикуму містить методичні вказівки для проведення лабораторних робіт з дисципліни «Технології розробки програмного забезпечення» у другому семестрі вивчення дисципліни (модуль 2) і за змістом відповідає робочій програмі дисципліни. Модуль 2 «MVC реалізація - AngularJS» розрахований на 36 годин лекційного матеріалу, 36 годин лабораторних робіт та 57 годин самостійної роботи. Модуль

містить 8 лабораторних робіт, які пов’язані з фреймворком для розробки web-додатків з відкритим вихідним кодом.

Кожна лабораторна робота містить такі елементи як мету, обладнання, теоретичні відомості, порядок виконання роботи, зміст звіту, запитання для самоконтролю, тривалість заняття. Звіт з лабораторної роботи оформляється відповідно до стандартів на оформлення технічної документації, які діють у навчальному закладі, повинен містити такі необхідні елементи: мету роботи, завдання, хід виконання роботи, одержані результати з графічною демонстрацією, текст програм з коментарями, висновки.

ЛАБОРАТОРНА РОБОТА 1.
ІНСТРУМЕНТИ ДЛЯ ПРИСКОРЕНОЇ РОЗРОБКИ НА ANGULAR.JS (BOWER , GRUNT, GULP)

Мета роботи: одержання практичних навиків у використанні Bower , Grunt, Gulp для побудови web-проекту і виконання задач.

Обладнання:

– проінстальований NPM;
– каталог з вихідним програмним кодом.

1.1 Теоретичні відомості

Підготовка до виконання завдання:

- встановити Node.JS.

Node.js це платформа для розробки серверних додатків на мові Javascript, що дозволяє швидко створювати додатки, що працюють в мережі. Використання Javascript одночасно на клієнтській і на стороні сервера дозволяє зробити розробку більш уніфікованою і спроектованою в рамках єдиної системи.

Встановлення під ОС Windows – відвідайте сайт http://nodejs.org. Для встановлення під ОС Ubuntu скористайтесь пакетним менеджером apt. Спочатку оновіть індекс пакетів, а потім встановіть дистрибутив з репозиторіїв:

```
sudo apt-get update
sudo apt-get install nodejs
```

Далі встановіть менеджер пакетів npm для Node.js:

```
sudo apt-get install npm
```

- завантажити з відповідного каталогу файл архіву conFusion.zip та розпакувати його в каталог з назвою conFusion. Ознайомитись з вмістом каталогу.

- встановлення Bower

На даний час існує величезна кількість фреймворків для розробки. У міру збільшення їх кількості в одному проекті стає складно їх усіх контролювати, для цих цілей існують менеджери пакетів. Вони спрощують установку сторонніх бібліотек і оновлення залежностей проекту.

Bower – популярний менеджер пакетів для клієнтського javascript, який встановлює пакети разом з їх залежностями. Встановлення Bower:

```
npm install -g bower
```

Відкрийте термінал (Git Bash), зайдіть в каталог conFusion. Введіть команду:

```
bower install
```

Це приведе до завантаження файлів Bootstrap, JQuery, Font Awesome і Angular в каталог bower_components.

Примітка. Для швидкого кодування на AngularJS встановіть для редактора Sublime додатковий пакет: https://dzone.com/articles/angularjs-how-code-quickly.

WEB TOOLS: GRUNT

1 Встановлення інтерфейсу Grunt (CLI) в командному режимі:

```
npm install -g grunt-cli
```

Вказана команда встановлює Grunt CLI глобально (можна використовувати у всіх проектах).

2 Створіть файл package.json в робочому каталозі проекту з наступним вмістом:

```
{
  "name": "conFusion",
  "private": true,
  "devDependencies": {

  },
  "engines": {
    "node": ">=0.10.0"
  }
}
```

3 Встановлення Grunt для використання в конкретному проекті: в робочому каталозі проекту введіть команду в командному рядку:

```
npm install grunt --save-dev
```

Наведена команда інсталює Grunt для використання в конкретному проекті.

4 Створення файлу Grunt: необхідно створити файл Grunt, який буде налаштовувати всі завдання, які буде запускати Grunt.

Необхідно створити файл з назвою Gruntfile.js в робочому каталозі проекту з наступним вмістом:

```
'use strict';

module.exports = function (grunt) {

  // Define the configuration for all the tasks
  grunt.initConfig({

  });

};
```

Такий вміст файлу налаштовує модуль Grunt, який готовий до включення завдань всередині функції, яка наведена вище.

5 Налаштування проекту: необхідно налаштувати файл menu.html для можливості оновлення імпорту CSS правил наступним чином:

```
        <!-- Bootstrap -->
<!-- build:css styles/main.css -->
    <link
href="../bower_components/bootstrap/dist/css/bootstrap.min.css"
rel="stylesheet">
    <link
href="../bower_components/bootstrap/dist/css/bootstrap-
theme.min.css" rel="stylesheet">
    <link        href="../bower_components/font-awesome/css/font-
awesome.min.css" rel="stylesheet">
    <link href="styles/bootstrap-social.css" rel="stylesheet">
    <link href="styles/mystyles.css" rel="stylesheet">
<!-- endbuild -->
```

Необхідно виокремити посилання спеціально відформатованими рядками. Аналогічно виділіть всі скрипти внизу сторінки. Увесь код Angular в подальшому буде переміщено в окремий файл.

```
<!-- build:js scripts/main.js -->
    <script
src="../bower_components/angular/angular.min.js"></script>
    <script src="scripts/app.js"></script>
<!-- endbuild -->
```

В каталозі app/scripts створіть файл з ім'ям app.js і помістіть в нього наступний код:

```
'use strict';
angular.module('confusionApp', [])

        .controller('menuController', function() {
                        this.tab = 1;
            this.filtText = '';
            var dishes=[
                        {
```

```
                            name:'Uthapizza',
                            image: 'images/uthapizza.png',
                            category: 'mains',
                            label:'Hot',
                            price:'4.99',
                            description:'A unique combination of
Indian Uthappam (pancake) and Italian pizza, topped with
Cerignola olives, ripe vine cherry tomatoes, Vidalia onion,
Guntur chillies and Buffalo Paneer.',
                              comment: ''
                           },
                           {
                              name:'Zucchipakoda',
                              image: 'images/zucchipakoda.png',
                              category: 'appetizer',
                              label:'',
                              price:'1.99',
                            description:'Deep    fried    Zucchini
coated with mildly spiced Chickpea flour batter accompanied with
a sweet-tangy tamarind sauce',
                              comment: ''
                           },
                           {
                              name:'Vadonut',
                              image: 'images/vadonut.png',
                              category: 'appetizer',
                              label:'New',
                              price:'1.99',
                            description:'A           quintessential
ConFusion experience, is it a vada or is it a donut?',
                              comment: ''
                           },
                           {
                              name:'ElaiCheese Cake',
                              image: 'images/elaicheesecake.png',
                            category: 'dessert',
                              label:'',
                           price:'2.99',
                            description:'A  delectable,  semi-sweet
New York Style Cheese Cake, with Graham cracker crust and spiced
with Indian cardamoms',
                              comment: ''
                           }
                           ];
            this.dishes = dishes;

            this.select = function(setTab) {
               this.tab = setTab;
               if (setTab === 2) {
                   this.filtText = "appetizer";
               }
               else if (setTab === 3) {
                   this.filtText = "mains";
               }
               else if (setTab === 4) {
                   this.filtText = "dessert";
               }
               else {
                   this.filtText = "";
               }
```

```
            };
            this.isSelected = function (checkTab) {
                return (this.tab === checkTab);
            };
        });
```

Такі зміни дозволять перемістити код JavaScript в окремий файл.

6 JSHint: завдання JSHint – перевіряти код JavaScript, вказувати на помилки і давати попередження про незначні невідповідності. Щоб зробити це, необхідно включити деякі модулі Grunt, які допомагають виконати завдання. Встановіть наступні модулі, виконавши наступні команди:

```
npm install grunt-contrib-jshint --save-dev
npm install jshint-stylish --save-dev
npm install time-grunt --save-dev
npm install jit-grunt --save-dev
```

Перший рядок встановлює модуль Grunt для JSHint, другий дає можливість виводу повідомлення JSHint в певному форматі, time-grunt модуль генерує статистику про те, скільки часу кожна задача виконувалась, і jit-grunt дозволяє включати необхідні завантажені модулі Grunt, коли це необхідно для виконання завдань.

Налаштування завдання JSHint в функції файлу Gruntfile.js:

```
  // Time how long tasks take. Can help when optimizing build
times
  require('time-grunt')(grunt);

  // Automatically load required Grunt tasks
  require('jit-grunt')(grunt);

  // Define the configuration for all the tasks
  grunt.initConfig({
          pkg: grunt.file.readJSON('package.json'),

    // Make sure code styles are up to par and there are no
obvious mistakes
    jshint: {
      options: {
        jshintrc: '.jshintrc',
        reporter: require('jshint-stylish')
      },
      all: {
        src: [
          'Gruntfile.js',
          'app/scripts/{,*/}*.js'
        ]
      }
    }
  });
```

```
  grunt.registerTask('build', [
    'jshint'
  ]);

  grunt.registerTask('default',['build']);
```

Завдання JSHint налаштоване протестувати всі файли JavaScript в каталозі app/scripts, і Gruntfile.js і генерувати звіти про помилки JS або попередження. Ми створили задачу Grunt з ім'ям `build`, яка запускає задачу jshint і ця задача визначена за замовчуванням.

7 Далі, створіть файл з ім'ям .jshintrc (Не забудьте “.” перед jshintrc) в робочому каталозі проекту, і запишіть в файл наступне:

```
{
  "bitwise": true,
  "browser": true,
  "curly": true,
  "eqeqeq": true,
  "esnext": true,
  "latedef": true,
  "noarg": true,
  "node": true,
  "strict": true,
  "undef": true,
  "unused": true,
  "globals": {
    "angular": false
  }
}
```

8 Запустіть на виконання завдання grunt з командного рядка:

```
grunt
```

9 Копіювання файлів і очищення каталогу Dist:

необхідно встановити модулі Grunt для копіювання файлів в каталог з назвою Dist, і очищення каталогу Dist при необхідності. Для цього встановіть наступні модулі Grunt:

```
npm install grunt-contrib-copy --save-dev
npm install grunt-contrib-clean --save-dev
```

Додайте наступний код у Gruntfile.js безпосередньо після конфігурації завдання JSHint:

```
    },
    copy: {
      dist: {
        cwd: 'app',
        src: [ '**','!styles/**/*.css','!scripts/**/*.js' ],
        dest: 'dist',
```

```
        expand: true
      },
      fonts: {
          files:[
              {
                  //for bootstrap fonts
                    expand: true,
                    dot: true,
                    cwd: 'bower_components/bootstrap/dist',
                    src: ['fonts/*.*'],
                    dest: 'dist'
                }, {
                    //for font-awesome
                    expand: true,
                    dot: true,
                    cwd: 'bower_components/font-awesome',
                    src: ['fonts/*.*'],
                    dest: 'dist'
                }
        ]
      }
    },
    clean: {
        build:{
            src: [ 'dist/']
        }
    }
```

Оновіть завдання Grunt build у файлі:

```
grunt.registerTask('build', [
  'clean',
  'jshint',
  'copy'
]);
```

Запустіть Grunt.

10 Підготовка каталогу Dist та файлів: необхідно інсталювати додаткові модулі Grunt – usemin, concat, cssmin, uglify і filerev :

```
npm install grunt-contrib-concat --save-dev
npm install grunt-contrib-cssmin --save-dev
npm install grunt-contrib-uglify --save-dev
npm install grunt-filerev --save-dev
npm install grunt-usemin --save-dev
```

Оновлення завдань в Gruntfile.js:

```
    },
    useminPrepare: {
        html: 'app/menu.html',
        options: {
            dest: 'dist'
        }
    },
      // Concat
    concat: {
        options: {
```

```
            separator: ';'
        },
        // dist configuration is provided by useminPrepare
        dist: {}
    },
      // Uglify
    uglify: {
        // dist configuration is provided by useminPrepare
        dist: {}
    },
    cssmin: {
        dist: {}
    },
      // Filerev
    filerev: {
        options: {
            encoding: 'utf8',
            algorithm: 'md5',
            length: 20
        },
        release: {
            // filerev:release hashes(md5) all assets (images,
js and css )
            // in dist directory
            files: [{
                src: [
                    'dist/scripts/*.js',
                    'dist/styles/*.css',
                ]
            }]
        }
    },
      // Usemin
      // Replaces all assets with their revved version in html
and css files.
      // options.assetDirs contains the directories for finding
the assets
      // according to their relative paths
    usemin: {
        html: ['dist/*.html'],
        css: ['dist/styles/*.css'],
        options: {
            assetsDirs: ['dist', 'dist/styles']
        }
    },
```

Оновлення налаштування jit-grunt для встановлення залежності завдання useminPrepare від пакету usemin:

```
require('jit-grunt')(grunt, {
  useminPrepare: 'grunt-usemin'
});
```

Оновлення завдання build Grunt :

```
  grunt.registerTask('build', [
    'clean',
    'jshint',
    'useminPrepare',
    'concat',
```

```
    'cssmin',
    'uglify',
    'copy',
    'filerev',
    'usemin'
  ]);
```

Запустіть Grunt, який створить каталог призначення (dist) з відповідною структурою для коректного розміщення web-сайту на сервері.

11 Завдання Watch, Connect і Serve: використання модулів Grunt для спостереження, підключення і автоматичного перезавантаження браузера, коли відстежувані файли оновлюються. Для цього встановіть наступні модулі:

```
npm install grunt-contrib-watch --save-dev
npm install grunt-contrib-connect --save-dev
```

Налаштуйте зв'язок і відстежування:

```
    watch: {
        copy: {
            files: [ 'app/**', '!app/**/*.css', '!app/**/*.js'],
            tasks: [ 'build' ]
        },
        scripts: {
            files: ['app/scripts/app.js'],
            tasks:[ 'build']
        },
        styles: {
            files: ['app/styles/mystyles.css'],
            tasks:['build']
        },
        livereload: {
            options: {
                livereload: '<%= connect.options.livereload %>'
            },
            files: [
                'app/{,*/}*.html',
                '.tmp/styles/{,*/}*.css',
                'app/images/{,*/}*.{png,jpg,jpeg,gif,webp,svg}'
            ]
      }
    },
    connect: {
      options: {
        port: 9000,
        // Change this to '0.0.0.0' to access the server from
outside.
        hostname: 'localhost',
        livereload: 35729
      },
      dist: {
        options: {
          open: true,
          base:{
               path: 'dist',
```

```
            options: {
                index: 'menu.html',
                maxAge: 300000
            }
          }
        }
      }
    },
```

Додайте відповідне завдання в файл Grunt:

```
grunt.registerTask('serve',['build','connect:dist','watch']);
```

Тепер, якщо ввести у командному рядку відповідну команду, буде побудований проект і відкриється web-сторінка в браузері, який використовується за замовчуванням. Також будуть відстежуватись файли в каталозі програми, і якщо ви оновите будь-який з них, буде перебудований проект і завантажена оновлена сторінка в браузері:

```
grunt serve
```

WEB TOOLS: GULP

12 Очищинення каталогу node_modules: удаліть всі файли, які містять в каталозі node_modules в каталозі проекту. Модулі Grunt, які містяться в даному каталозі не будуть використовуватись.

13 Ініціалізація файлу package.json: оновіть вміст файлу package.json:

```
{
  "name": "conFusion",
  "private": true,
  "devDependencies": {

  },
  "engines": {
    "node": ">=0.10.0"
  }
}
```

14 Встановлення Gulp глобально:

```
npm install -g gulp
```

15 Встановлення Gulp в проекті з каталогу проекту:

```
npm install gulp --save-dev
```

16 Встановлення плагінів:

```
npm install jshint gulp-jshint jshint-stylish gulp-imagemin
gulp-concat gulp-uglify gulp-minify-css gulp-usemin gulp-cache
gulp-changed gulp-rev gulp-rename gulp-notify  browser-sync del
--save-dev
```

17 Створення файлу Gulp з завданнями, які будуть запускатись при запуску Gulp: створіть файл gulpfile.js в каталозі проекту.

Для завантаження встановлених плагінів помістіть код у створений файл:

```
var gulp = require('gulp'),
    minifycss = require('gulp-minify-css'),
    jshint = require('gulp-jshint'),
    stylish = require('jshint-stylish'),
    uglify = require('gulp-uglify'),
    usemin = require('gulp-usemin'),
    imagemin = require('gulp-imagemin'),
    rename = require('gulp-rename'),
    concat = require('gulp-concat'),
    notify = require('gulp-notify'),
    cache = require('gulp-cache'),
    changed = require('gulp-changed'),
    rev = require('gulp-rev'),
    browserSync = require('browser-sync'),
    del = require('del');
```

Додавання завдання Gulp для JSHint , очистки і завдання за замовчуванням:

```
gulp.task('jshint', function() {
  return gulp.src('app/scripts/**/*.js')
  .pipe(jshint())
  .pipe(jshint.reporter(stylish));
});

// Clean
gulp.task('clean', function() {
    return del(['dist']);
});

// Default task
gulp.task('default', ['clean'], function() {
    gulp.start('usemin', 'imagemin','copyfonts');
});
```

Додати в код завдання usemin, imagemin і copyfonts:

```
gulp.task('usemin',['jshint'], function () {
  return gulp.src('./app/menu.html')
      .pipe(usemin({
        css:[minifycss(),rev()],
        js: [uglify(),rev()]
      }))
      .pipe(gulp.dest('dist/'));
});

// Images
gulp.task('imagemin', function() {
  return del(['dist/images']), gulp.src('app/images/**/*')
    .pipe(cache(imagemin({  optimizationLevel:  3,  progressive:
true, interlaced: true })))
    .pipe(gulp.dest('dist/images'))
```

```
    .pipe(notify({ message: 'Images task complete' }));
});

gulp.task('copyfonts', ['clean'], function() {
   gulp.src('./bower_components/font-
awesome/fonts/**/*.{ttf,woff,eof,svg}*')
   .pipe(gulp.dest('./dist/fonts'));

gulp.src('./bower_components/bootstrap/dist/fonts/**/*.{ttf,woff
,eof,svg}*')
   .pipe(gulp.dest('./dist/fonts'));
});
```

Додавання коду для задач відстежування і browserSync:

```
// Watch
gulp.task('watch', ['browser-sync'], function() {
  // Watch .js files

gulp.watch('{app/scripts/**/*.js,app/styles/**/*.css,app/**/*.ht
ml}', ['usemin']);
      // Watch image files
  gulp.watch('app/images/**/*', ['imagemin']);

});

gulp.task('browser-sync', ['default'], function () {
   var files = [
      'app/**/*.html',
      'app/styles/**/*.css',
      'app/images/**/*.png',
      'app/scripts/**/*.js',
      'dist/**/*'
   ];

   browserSync.init(files, {
      server: {
         baseDir: "dist",
         index: "menu.html"
      }
   });
        // Watch any files in dist/, reload on change
  gulp.watch(['dist/**']).on('change', browserSync.reload);
    });
```

Збережіть Gulp file.

18 Запуск завдань Gulp:

```
gulp
```

19 Використання BrowserSync і Watch: для виконання завдань виконайте наступну команду в командному рядку:

```
gulp watch
```

Спробуйте відредагувати файл html і подивитись на зміни на сторінці.

1.2 Порядок виконання роботи

1 Виконайте пункти 1-11 з теоретичних відомостей.

2 Налаштуйте завдання Grunt для створення каталогу app_dist з відповідною структурою і файлом dishdetail.html для коректного розміщення web-сторінки на сервері.

3 Налаштуйте Grunt для спостереження за змінами файлу dishdetail.html.

4 Виконайте пункти 12-19 з теоретичних відомостей.

5 Налаштуйте завдання Gulp для створення каталогу app_dist з відповідною структурою і файлом dishdetail.html для коректного розміщення web-сторінки на сервері.

6 Налаштуйте Gulp для спостереження за змінами файлу dishdetail.html.

1.3 Зміст звіту

Звіт повинен містити:

- титульний аркуш;
- мету роботи і завдання;
- покроковий опис роботи, команди операційної системи для встановлення пакетів, копії екранів з результатами виконаної роботи; вихідний код створених файлів *.js, *.html.
- висновки.

Запитання для самоконтролю:

Тривалість заняття: 4 год.

ЛАБОРАТОРНА РОБОТА 2.
РОЗРОБЛЕННЯ ДОДАТКУ З ФРОНТЕНДОМ НА ANGULAR.JS

Мета роботи: одержати практичні навики у використанні директив AngularJS (ng-app, ng-init, ng-model і ng-repeat) при розробці web-додатку; одержати практичні навики у використанні модулів та контролерів при формуванні додатку Angular.

Обладнання:

– доступ до мережі internet;

– ПК IBM PC x86 CPU з встановленою операційною системою.

2.1 Теоретичні відомості

AngularJS – це фреймворк для web-додатків з відкритим вихідним кодом. Підтримується Google і співтовариством окремих розробників і корпорацій для вирішення проблем, що виникають при розробці односторінкових додатків. Він націлений на спрощення розробки і тестування подібних додатків, пропонуючи фреймворк для клієнтської частини архітектури model-view-controller (MVC, модель-представлення-контролер) і model-view-view-model (MVVM, модель-представлення-представлення-модель) разом з компонентами, які застосовуються в насичених інтернет-додатках.

Бібліотека AngularJS починає роботу з читання HTML-сторінки, в яку впроваджені додаткові атрибути користувацьких тегів. Angular інтерпретує ці атрибути у вигляді директив для зв'язування вхідних і вихідних областей сторінки з моделлю, яка представлена стандартними змінними JavaScript. Значення цих змінних JavaScript можна встановити вручну в коді або одержати з статичних або динамічних ресурсів JSON.

1 Додавання Angular у файл: для додавання можливості використання Angular сторінкою необхідно включити файли Angular JavaScript у файл *.html шляхом додавання наступного коду в кінці сторінки (безпосередньо перед закриваючим тегом body) :

```
<script
src="../bower_components/angular/angular.min.js"></script>
```

Тут використовуються файли Angular, які завантажив Bower.

2 Для конфігурування web-сторінки для додатку Angular необхідно додати директиву ng-app до тегу **<html>** , наприклад:

```
<html lang="en" ng-app>
```

3 Додавання даних з використанням директиви ng-init:

```
<div class="row row-content"
      ng-init="
         dish=
         {
           name:'Uthapizza',
           image: 'images/uthapizza.png',
           category: 'mains',
           label:'Hot',
           price:'4.99',
           description:'A unique combination of Indian Uthappam
(pancake) and Italian pizza, topped with Cerignola olives, ripe
vine cherry tomatoes, Vidalia onion, Guntur chillies and Buffalo
Paneer.',
           comment: ''
        }">
```

Наведений фрагмент коду включає об’єкт JavaScript в код.

4 Звертання до властивостей об’єкту JavaScript відбувається з використанням виразів Angular з подвійними фігурними дужками {{ ... }}:

```
<div class="col-xs-12">
    <div class="media">
        <div class="media-left media-middle">
            <a href="#">
            <img class="media-object img-thumbnail"
             ng-src={{dish.image}} alt="Uthappizza">
            </a>
        </div>
        <div class="media-body">
            <h2 class="media-heading">{{dish.name}}
             <span              class="label              label-
danger">{{dish.label}}</span>
              <span          class="badge">{{dish.price          |
currency}}</span></h2>
            <p>{{dish.description}}</p>
        </div>
    </div>
</div>
```

В наведеному коді використано фільтр Angular для відображення форматування ціни.

5 Зв’язування: використання зв’язування для оновлення інформації на сторінці при введенні її в поле input:

```
<div class="col-xs-12">
    <div class="media">
            . . .
            <p>{{dish.description}}</p>
            <p>Comment: {{dish.comment}}</p>
            <p>Type your comment:
             <input type="text" ng-model="dish.comment"></p>
        </div>
    </div>
</div>
```

6 Об’єкт Array і ng-repeat: оновимо вміст директиви ng-init додаванням масиву об’єктів JavaScript:

```
<div class="row row-content"
               ng-init="
                 dishes=[
                 {
                   name:'Uthapizza',
                   image: 'images/uthapizza.png',
                   category: 'mains',
                   label:'Hot',
                   price:'4.99',
                   description:'A unique combination of Indian
Uthappam (pancake) and Italian pizza, topped with Cerignola
olives, ripe vine cherry tomatoes, Vidalia onion, Guntur
chillies and Buffalo Paneer.',
                   comment: ''
                 },
                 {
                    name:'Zucchipakoda',
                    image: 'images/zucchipakoda.png',
                    category: 'appetizer',
                    label:'',
                    price:'1.99',
                    description:'Deep fried Zucchini coated
with mildly spiced Chickpea flour batter accompanied with a
sweet-tangy tamarind sauce',
                    comment: ''
                  },
                  {
                    name:'Vadonut',
                    image: 'images/vadonut.png',
                    category: 'appetizer',
                     label:'New',
                     price:'1.99',
                     description:'A quintessential ConFusion
experience, is it a vada or is it a donut?',
                     comment: ''
                   },
                   {
                     name:'ElaiCheese Cake',
                     image: 'images/elaicheesecake.png',
                     category: 'dessert',
                     label:'',
                     price:'2.99',
                     description:'A delectable, semi-sweet New
York Style Cheese Cake, with Graham cracker crust and spiced
with Indian cardamoms',
```

```
                    comment: ''
                  }
                ]">
```

Оновимо код шляхом створення списку об’єктів, які будуть формуватись з масиву об’єктів JavaScript з використанням директиви ng-repeat:

```
<div class="col-xs-12">
   <ul class="media-list">
    <li class="media" ng-repeat="dish in dishes">
        <div class="media-left media-middle">
            <a href="#">
            <img class="media-object img-thumbnail"
             ng-src={{dish.image}} alt="Uthappizza">
            </a>
        </div>
        <div class="media-body">
            <h2 class="media-heading">{{dish.name}}
             <span              class="label              label-
danger">{{dish.label}}</span>
             <span          class="badge">{{dish.price          |
currency}}</span></h2>
            <p>{{dish.description}}</p>
            <p>Comment: {{dish.comment}}</p>
            <p>Type your comment:
             <input type="text" ng-model="dish.comment"></p>
        </div>
    </li>
</ul>
</div>
```

7 Створення модуля Angular: оновити ng-app ім'ям для модуля, який визначатиме додаток:

```
<html lang="en" ng-app="confusionApp">
```

Це означає, що необхідно визначити модуль з ім’ям confusionApp: перед закриваючим тегом body додаємо код для створення модуля:

```
<script>

    var app = angular.module('confusionApp',[]);

</script>
```

8 Додаємо контроллер до app:

```
<script>
    var app = angular.module('confusionApp',[]);
    app.controller('menuController', function() {

    });

</script>
```

Переміщуємо масив об’єктів JavaScript з ng-init до контролера. Удаляємо ng-init з div і додаємо наступний код до Controller:

```
var dishes=[
             {
                name:'Uthapizza',
                image: 'images/uthapizza.png',
                category: 'mains',
                label:'Hot',
                price:'4.99',
                description:'A unique combination of Indian Uthappam (pancake) and Italian pizza, topped with Cerignola olives, ripe vine cherry tomatoes, Vidalia onion, Guntur chillies and Buffalo Paneer.',
                comment: ''
            },
            {
                name:'Zucchipakoda',
                image: 'images/zucchipakoda.png',
                category: 'appetizer',
                label:'',
                price:'1.99',
                description:'Deep fried Zucchini coated with mildly spiced Chickpea flour batter accompanied with a sweet-tangy tamarind sauce',
                comment: ''
            },
            {
                name:'Vadonut',
                image: 'images/vadonut.png',
                category: 'appetizer',
                label:'New',
                price:'1.99',
                description:'A quintessential ConFusion experience, is it a vada or is it a donut?',
                comment: ''
            },
            {
                name:'ElaiCheese Cake',
                image: 'images/elaicheesecake.png',
                category: 'dessert',
                label:'',
                price:'2.99',
                description:'A delectable, semi-sweet New York Style Cheese Cake, with Graham cracker crust and spiced with Indian cardamoms',
                comment: ''
            }
            ];

this.dishes = dishes;
```

9 Приєднання контролера: змінюємо клас row шляхом додавання контролера до div:

```
<div class="row row-content" ng-controller="menuController as menuCtrl">
```

Вказана дія додасть menuController як контролер для div і визначить скорочення для контролера як menuCtrl.

Оновлення елементів списку:

```
<li class="media" ng-repeat="dish in menuCtrl.dishes">
```

Web-сторінка не буде відображати ніяких змін, за винятком того, що тепер код перенесено в контролер.

2.2 Порядок виконання роботи

Збережіть у файл dishdetail.html нижче наведений вихідний код сторінки:

```
<!DOCTYPE html>
<html lang="en" ng-app="confusionApp">

<head>
     <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-
scale=1">
    <!-- The above 3 meta tags *must* come first in the head;
any other head
         content must come *after* these tags -->
    <title>Ristorante Con Fusion: Menu</title>
        <!-- Bootstrap -->
    <link
href="../bower_components/bootstrap/dist/css/bootstrap.min.css"
rel="stylesheet">
    <link
href="../bower_components/bootstrap/dist/css/bootstrap-
theme.min.css" rel="stylesheet">
    <link       href="../bower_components/font-awesome/css/font-
awesome.min.css" rel="stylesheet">
    <link href="styles/bootstrap-social.css" rel="stylesheet">
    <link href="styles/mystyles.css" rel="stylesheet">

    <!-- HTML5 shim and Respond.js for IE8 support of HTML5
elements and media queries -->
    <!-- WARNING: Respond.js doesn't work if you view the page
via file:// -->
    <!--[if lt IE 9]>
      <script
src="https://oss.maxcdn.com/html5shiv/3.7.2/html5shiv.min.js"></
script>
      <script
src="https://oss.maxcdn.com/respond/1.4.2/respond.min.js"></scri
pt>
    <![endif]-->
</head>

<body>

    <div class="container">
        <div class="row row-content">
            <div class="col-xs-12">
```

```
                <p>Put the dish details here</p>
            </div>
            <div class="col-xs-9 col-xs-offset-1">
                <p>Put the comments here</p>
            </div>
        </div>

    </div>

    <script
src="../bower components/angular/angular.min.js"></script>
    <script>

        var app = angular.module('confusionApp',[]);

        app.controller('dishDetailController', function() {

            var dish={
                        name:'Uthapizza',
                        image: 'images/uthapizza.png',
                        category: 'mains',
                        label:'Hot',
                        price:'4.99',
                        description:'A unique combination of
Indian Uthappam (pancake) and Italian pizza, topped with
Cerignola olives, ripe vine cherry tomatoes, Vidalia onion,
Guntur chillies and Buffalo Paneer.',
                         comments: [
                             {
                                 rating:5,
                                 comment:"Imagine    all    the
eatables, living in conFusion!",
                                 author:"John Lemon",
                                 date:"2012-10-
16T17:57:28.556094Z"
                             },
                             {
                                 rating:4,
                                 comment:"Sends    anyone    to
heaven, I wish I could get my mother-in-law to eat it!",
                                 author:"Paul McVites",
                                 date:"2014-09-
05T17:57:28.556094Z"
                             },
                             {
                                 rating:3,
                                 comment:"Eat   it,   just   eat
it!",
                                 author:"Michael Jaikishan",
                                 date:"2015-02-
13T17:57:28.556094Z"
                             },
                             {
                                 rating:4,
                                 comment:"Ultimate,    Reaching
for the stars!",
                                 author:"Ringo Starry",
                                 date:"2013-12-
02T17:57:28.556094Z"
                             },
```

```
                                    {
                                        rating:2,
                                        comment:"It's  your  birthday,
we're gonna party!",
                                        author:"25 Cent",
                                        date:"2011-12-
02T17:57:28.556094Z"
                                    }

                                ]
                        };

                this.dish = dish;

            });

        </script>

</body>

</html>
```

Завдання полягає у формуванні сторінки з використанням Angular для відображення інформації про страву та коментарі клієнтів про блюдо. Необхідно виконати наступні завдання по оновленню сторінку:

1 показати деталі про блюдо;

2 відображення списку коментарів;

3 надати можливість клієнту сортування коментарів за автором, датою або рейтингом.

Детальні інструкції по кожному завданні наведені нижче.

Завдання 1

Відображення інформації про страву на сторінці. Повинно бути відображено в першому стовпці DIV усередині row div. Щоб зробити це, вам потрібно виконати наступні дії:

- налаштуйте row для використання dishDetailController;
- використовуйте Bootstrap Media Object для відображення інформації про страву.

Завдання 2

У цьому завданні необхідно відобразити всі коментарі про блюдо з об'єкту JavaScript як список коментарів під описом блюда на сторінці. Коментарі повинні бути відображені у другому стовпці DIV усередині row div. Щоб зробити це, вам потрібно виконати наступні дії:

- використовуйте клас blockquote Bootstrap для форматування коментарів. Автор і інформація про дату повинні бути в footer blockquote;
- поле дати повинні бути відформатовані за допомогою Angular date filter;
- коментарі формуються на основі перебору елементів масиву коментарів з використанням відповідної ng- директиви.

Завдання 3

У цьому завданні ви додасте фільтр для секції коментарів, так що коментарі могли бути впорядковані відповідно до обраних критеріїв користувача. Використайте для цього фільтр OrderBy Angular. Щоб зробити це, вам потрібно виконати наступні дії:

- налаштуйте поле введення, яке дозволяє користувачеві ввести свої критерії сортування: автор, дата і рейтинг. Вони можуть бути з префіксом “-” (знак для позначення зворотного порядку);
- налаштуйте OrderBy фільтр для коментарів, так щоб коментарі сортувались у відповідності з обраним критерієм.

Результат виконання роботи має відповідати рисунку 2.1.

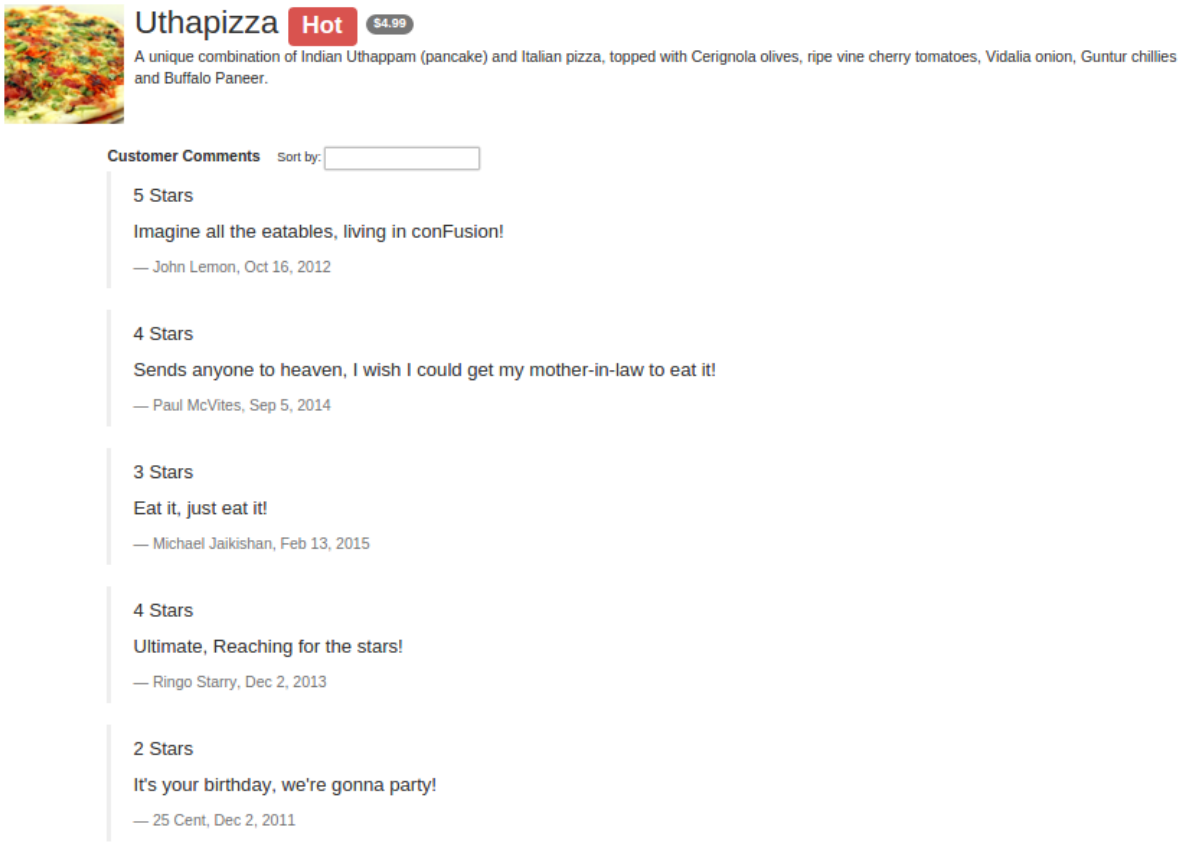

Рисунок 2.1 – Відображення сторінки після виконання завдань лабораторної роботи

2.3 Зміст звіту

Звіт має містити:

– титульний аркуш;

– мету роботи і завдання;

– покроковий опис роботи, програмний код для реалізації кожного завдання, копії екранів з результатами виконаної роботи.

– висновки.

Запитання для самоконтролю:

Тривалість заняття: 6 год.

ЛАБОРАТОРНА РОБОТА 3.
ФОРМИ ANGULAR, ВАЛІДАЦІЯ ФОРМ

Мета роботи: отримати навики у використанні форм у додатку Angular з можливістю валідації.

Обладнання:

– доступ до мережі internet;

– ПК IBM PC x86 CPU з встановленою операційною системою.

3.1 Теоретичні відомості

1 Збережіть файл contactus.html з каталогу лабораторної роботи в каталог *conFusion/app*. Відкрийте файл для редагування.

Підтримка форм в Angular:

2 оновіть тег <html> для можливості використання додатку Angular як показано нижче

```
<html lang="en" ng-app="confusionApp">
```

3 Оновіть контейнер div в тілі сторінки:

```
<div class="container" ng-controller="ContactController">
```

4 У формі оновіть закриваючий тег div як показано нижче:

```
<div class="col-xs-12 col-sm-9" ng-controller="FeedbackController">
```

Додавання контролерів:

5 Відкрийте файл *app.js* і в нижній частині цієї сторінки додайте два нових контролери, які ми використовували на сторінці, додавши наступний код:

```
.controller('ContactController', ['$scope', function($scope) {

            $scope.feedback = {mychannel:"", firstName:"", lastName:"",
                               agree:false, email:"" };
        }])

        .controller('FeedbackController', ['$scope', function($scope) {

        }]);
```

Налаштування двостороннього зв'язування:

6 Оновіть поле вводу firstName як показано нижче:

```
<div class="form-group">
     <label for="firstname" class="col-sm-2 control-label">First Name</label>
     <div class="col-sm-10">
     <input type="text" class="form-control" id="firstname" name="firstname"
placeholder="Enter First Name"
     ng-model="feedback.firstName" required>
     </div>
```

```
</div>
```

7 Оновіть поле вводу lastName як показано:

```
<div class="form-group">
     <label for="lastname" class="col-sm-2 control-label">Last Name</label>
     <div class="col-sm-10">
          <input      type="text"      class="form-control"      id="lastname"
placeholder="Enter Last Name"
      ng-model="feedback.lastName" required>
      </div>
</div>
```

8 Оновіть номери телефонів як показано нижче:

```
                    <div class="form-group">
                        <label    for="telnum"    class="col-sm-2    control-
label">Contact Tel.</label>
                        <div class="col-xs-5 col-sm-4 col-md-3">
                            <div class="input-group">
                                <div class="input-group-addon">(</div>
                                    <input   type="tel"   class="form-control"
placeholder="Area  code"                                                    ng-
model="feedback.tel.areaCode">
                                <div class="input-group-addon">)</div>
                            </div>
                        </div>
                        <div class="col-xs-7 col-sm-6 col-md-7">
                                    <input   type="tel"   class="form-control"
id="telnum" name="telnum" placeholder="Tel. number"
                                    ng-model="feedback.tel.number">
                        </div>
                    </div>
```

9 Оновіть поля електронної адреси як показано нижче:

```
                    <div class="form-group">
                        <label    for="emailid"    class="col-sm-2    control-
label">Email</label>
                        <div class="col-sm-10">
                            <input      type="email"      class="form-control"
id="emailid" name="emailid" placeholder="Email"
                            ng-model="feedback.email" required>
                        </div>
                    </div>
```

10 Оновіть поле прапорця (checkbox):

```
                   <div class="checkbox col-sm-5 col-sm-offset-2">
                        <label class="checkbox-inline">
                            <input type="checkbox"
                            ng-model="feedback.agree">
                            <strong>May we contact you?</strong>
                        </label>
                    </div>
```

11 Оновіть поле вводу тексту:

```
                    <div class="form-group">
                        <label    for="feedback"    class="col-sm-2    control-
label">Your Feedback</label>
                        <div class="col-sm-10">
                            <textarea class="form-control" rows="12"
                                ng-model="feedback.comments">
                            </textarea>
                        </div>
                    </div>
```

12 Оновіть малу колонку праворуч від форми зворотного зв'язку в такий спосіб:

```
            <div class="col-xs-12 col-sm-3">
             <h3>Your Current Feedback:</h3>
             <p>{{feedback.firstName}}   {{feedback.lastName   |   uppercase
}}</p>
             <p>Contact           Tel.:           ({{feedback.tel.areaCode}})
{{feedback.tel.number}}</p>
             <p>Contact Email: {{feedback.email}}</p>
             <p             ng-show="feedback.agree">Contact             by:
{{feedback.mychannel}}</p>
             <p>Comments: {{feedback.comments}}</p>
            </div>
```

Валідація Angular форм:

13 Оновіть тег <form> як показано нижче:

```
               <form    class="form-horizontal"    name="feedbackForm"    ng-
submit="sendFeedback()" novalidate>
```

14 Далі, оновіть firstName з form group як показано нижче:

```
                    <div   class="form-group"   ng-class="{   'has-error'   :
feedbackForm.firstname.$error.required   &&   !feedbackForm.firstname.$pristine
}">

                        . . .

                        <span  ng-show="feedbackForm.firstname.$error.required
&&  !feedbackForm.firstname.$pristine"  class="help-block">Your  First  name  is
required.</span>
                        </div>
                    </div>
```

15 Оновіть lastname з form group як показано нижче:

```
                    <div   class="form-group"   ng-class="{   'has-error'   :
feedbackForm.lastname.$error.required && !feedbackForm.lastname.$pristine }">

                        . . .

                        <span   ng-show="feedbackForm.lastname.$error.required
&&  !feedbackForm.lastname.$pristine"  class="help-block">Your  Last  name  is
required.</span>
                        </div>
                    </div>
```

16 Потім оновіть поле електронної адреси:

```
                    <div  class="form-group"    ng-class="{  'has-error  has-
feedback'  :  feedbackForm.emailid.$invalid  &&  !feedbackForm.emailid.$pristine
}">

                        . . .

                        <span    ng-show="feedbackForm.emailid.$invalid    &&
!feedbackForm.emailid.$pristine"   class="glyphicon   glyphicon-remove   form-
control-feedback" aria-hidden="true"></span>
                        <span ng-show="(feedbackForm.emailid.$invalid ||
                                        feedbackForm.emailid.$error.required)
&&
                                        !feedbackForm.emailid.$pristine"
                              class="help-block">Enter    a    valid    email
address.</span>
                        </div>
                    </div>
```

17 Оновіть поле прапорця і вибору з form group як показано нижче:

```
                    <div  class="form-group"     ng-class="{  'has-error'  :
invalidChannelSelection }">                          <div class="checkbox col-
sm-5 col-sm-offset-2">
                            <label class="checkbox-inline">
                                <input type="checkbox"
                                ng-model="feedback.agree">
                                <strong>May we contact you?</strong>
                            </label>
                        </div>
                        <div     class="col-sm-3     col-sm-offset-1"     ng-
show="feedback.agree">
                            <select class="form-control"
                               ng-model="feedback.mychannel"
                               ng-options="channel.value as channel.label for
channel in channels">
                                <option value="">Tel. or Email?</option>
                            </select>
                        <span  ng-show="invalidChannelSelection"  class="help-
block">Select an option.</span>
                        </div>
                    </div>
```

18 Оновіть кнопку у формі відповідно:

```
                            <button type="submit" class="btn btn-primary" ng-
disabled="feedbackForm.$invalid">Send Feedback</button>
```

19 Збережіть файл *contactus.html* і відкрийте файл *app.js*. Оновіть код контролера як показано нижче:

```
.controller('ContactController', ['$scope', function($scope) {
            $scope.feedback   =   {mychannel:"",   firstName:"",   lastName:"",
agree:false, email:"" };
                        var   channels   =   [{value:"tel",   label:"Tel."},
{value:"Email",label:"Email"}];
                        $scope.channels = channels;
            $scope.invalidChannelSelection = false;
                            }])
        .controller('FeedbackController', ['$scope', function($scope) {
                        $scope.sendFeedback = function() {
                            console.log($scope.feedback);
                            if          ($scope.feedback.agree          &&
($scope.feedback.mychannel == "")&& !$scope.feedback.mychannel) {
                    $scope.invalidChannelSelection = true;
                    console.log('incorrect');
                }
                else {
                    $scope.invalidChannelSelection = false;
                    $scope.feedback     =     {mychannel:"",     firstName:"",
lastName:"",
                                        agree:false, email:"" };
                    $scope.feedback.mychannel="";

                    $scope.feedbackForm.$setPristine();
                    console.log($scope.feedback);
                }
            };
        }]);
```

20 Збережіть файл і дослідіть поведінку web-сторінки.

3.2 Порядок виконання роботи

Завдання 1. У цьому завданні необхідно сформувати форму для користувачів з можливістю введення зауваження. Необхідно виконати наступні дії:

- налаштуйте форму з трьома полями: ім'я, рейтинг і коментар;
- використайте об'єкт JavaScript в контролері для стеження за інформацією. Рейтинг повинен бути встановлений на значення 5 за замовчуванням. Використовуйте радіо переключатель для визначення рейтингу в формі;
- використовуйте двостороннє зв'язування даних між полями форми і об'єкта JavaScript.

Завдання 2. У цьому завданні, необхідно дозволити перевірку форми, використовуючи Angular підтримку для перевірки форм. Необхідно виконати наступні дії:

- ім’я та коментарі є обов'язковими для заповнення полями і користувач повинення ввести відповідну інформацію в обох полях;
- кнопка відправки повинна бути відключена, поки користувач не заповнить всі необхідні поля;
- користувач повинен бути попереджений, підсвічуючи недопустимі поля червоним кольором, повідомлення повинні відображатись в нижній частині поля.

Завдання 3. Виконання завдання дозволить в режимі реального часу переглядати коментарів користувача на web-сторінці. Попередній перегляд повинен відображатись в тому ж форматі, що і звичайні коментарі. Для цього необхідно виконати наступні завдання:

- показати попередній перегляд трохи вище форми, використовуючи той же формат, як і звичайні коментарі на сторінці;
- відображення попереднього перегляду, тільки якщо користувач ввів коректну інформацію в форму, а стан форми відрізняється від стану за замовчуванням;
- при відправлення коректного коментаря, коментар повинен приєднатися до існуючих коментарів на сторінці. Дата коментаря повинна бути встановлена автоматично при подачі форми.

Оформити звіт.

3.3 Зміст звіту

Звіт має містити:

- титульний аркуш;
- мету роботи і завдання;
- покроковий опис роботи, програмний код для реалізації кожного завдання, копії екранів з результатами виконаної роботи.
- висновки.

Запитання для самоконтролю:

Тривалість заняття: 4 год.

ЛАБОРАТОРНА РОБОТА 4.
МАРШРУТИЗАЦІЯ НА ANGULAR.JS

Мета роботи: отримати навики у побудові шаблонів з використанням HTML; навчитись включати Angular представлення в сторінку з використанням директиви ngInclude; навчитись будувати односторінковий додаток (single page application, SPA).

Обладнання:

– доступ до мережі internet;

– ПК IBM PC x86 CPU з встановленою операційною системою.

4.1 Теоретичні відомості

Маршрутизація в web-додатку.

Як правило, web-додатки використовують адреси, що читаються, і описують зміст, який за ними ховається. Типовим прикладом може бути клацання по посиланню на головну сторінку: це означає, що дія буде виконана на стороні сервера, потім результат буде переданий в інше представлення на стороні клієнта, наприклад, після взаємодії в корені web-додатку (/ або index.html), відбуваються зміни в адресному рядку браузера.

У AngularJS можна використовувати модуль ngRoute для маршрутизації. Особливості його застосування:

- щоб використовувати модуль ngRoute необхідно включити angular-route.js в додаток після скрипта angular.js;

- потрібну маршрутизацію необхідно налаштувати всередині модуля, щоб було простіше визначити модуль в окремому файлі script.js;

- необхідно вказати однакове ім'я для ng-app (в HTML-файлі, що містить додаток Angular) і при визначенні модуля.

UI-Router маршрутизація фреймворка для AngularJS створена командою AngularUI. Вона забезпечує інший підхід, ніж ngRoute, і змінює стан додатку, а не тільки URL маршрут. При такому підході, представлення і маршрути не прив'язані до URL сайту. Таким чином, можна змінити деталі сайту за допомогою маршрутизації, навіть якщо URL не змінюється.

4.2 Порядок виконання роботи

1 Angular шаблони (Templates): збережіть наведений нижче код у файл index.html у каталог app. Переконайтесь, що тег <html> містить директиву ngApp:

```
<html lang="en" ng-app="confusionApp">
```

Відредагуйте сторінки menu.html, contactus.html і dishdetail.htm шляхом видалення інформації заголовка і підключення скриптів з низу сторінок. Збережіть тільки зміст, вкладений в контейнер div.

Тепер переходимо до index.html і включіть наступне між заголовком (header) і колонтитулом (footer):

```
<ng-include src="'menu.html'"></ng-include>
```

Angular ngRoute і односторінкові додатки (SPAs)

1 застосуйте Bower для встановлення angular-маршруту – введіть у командному рядку:

```
bower install angular-route -S
```

2 Налаштуйте index.html: додайте angular-route.js до index.html:

```
<script src="../bower_components/angular-route/angular-route.min.js"></script>
```

3 Замініть директиву ngInclude директивою ngView:

```
<ng-view></ng-view>
```

Збережіть файл index.html.

4 Налаштуйте ngRoute в app.js: відкрийте app.js і використайте включення залежностей для включення ngRoute:

```
angular.module('confusionApp', ['ngRoute'])
```

Додайте наступний код для налаштування маршруту:

```
.config(function($routeProvider) {
        $routeProvider
            // route for the contactus page
            .when('/contactus', {
                templateUrl : 'contactus.html',
                controller  : 'ContactController'
            })
            // route for the menu page
            .when('/menu', {
                templateUrl : 'menu.html',
                controller  : 'MenuController'
            })
            // route for the dish details page
            .when('/menu/:id', {
```

```
            templateUrl : 'dishdetail.html',
            controller  : 'DishDetailController'
        })
        .otherwise('/contactus');
})
```

5 Оновіть services.js: відкрийте файл services.js і налаштуйте об”єкт dishes для включення поля id для кожного об”єкту dish:

```
        var dishes=[
                    {
                     _id:0,
                     name:'Uthapizza',

            . . .

                    },
                    {
                     _id:1,
                     name:'Zucchipakoda',

            . . .

                    }
             ];
```

6 Налаштуйте menu.html: відкрийте файл menu.html і оновіть посилання на рисунки відповідно:

```
                <div class="media-left media-middle">
                    <a ng-href="#/menu/{{dish._id}}">
                    <img class="media-object img-thumbnail"
                    ng-src={{dish.image}} alt="Uthappizza">
                    </a>
                </div>
```

7 Налаштуйте DishDetailController: відкрийте файл controllers.js і оновіть DishDetailController:

```
        .controller('DishDetailController', ['$scope',
'$routeParams', 'menuFactory', function($scope, $routeParams,
menuFactory) {

            var dish=
menuFactory.getDish(parseInt($routeParams.id,10));
$scope.dish = dish;
                }])
```

Збережіть і перегляньте сторінку.

Angular UI-Router для односторінкових додатків (Single Page Applications)

8 Використайте Bower для інсталювання angular-ui-router шляхом введення наступного в командному рядку:

```
bower install angular-ui-router -S
```

9 Налаштуйте UI router:

```
.config(function($stateProvider, $urlRouterProvider) {
        $stateProvider
                    // route for the home page
            .state('app', {
                url:'/',
                views: {
                    'header': {
                        templateUrl : 'views/header.html'
                    },
                    'content': {
                        template : '<h1>To be Completed</h1>',
                        controller  : 'IndexController'
                    },
                    'footer': {
                        templateUrl : 'views/footer.html'
                    }
                }
            })
                    // route for the aboutus page
            .state('app.aboutus', {
                url:'aboutus',
                views: {
                    'content@': {
                        template: '<h1>To be Completed</h1>',
                        controller  : 'AboutController'
                   }
                }
            })
                    // route for the contactus page
            .state('app.contactus', {
                url:'contactus',
                views: {
                    'content@': {
                        templateUrl : 'views/contactus.html',
                        controller  : 'ContactController'
                     }
                }
            })

            // route for the menu page
            .state('app.menu', {
                url: 'menu',
                views: {
                    'content@': {
                        templateUrl : 'views/menu.html',
                        controller  : 'MenuController'
                    }
                }
            })

            // route for the dishdetail page
            .state('app.dishdetails', {
                url: 'menu/:id',
```

```
                views: {
                    'content@': {
                        templateUrl : 'views/dishdetail.html',
                        controller  : 'DishDetailController'
                    }
                }
            });
            $urlRouterProvider.otherwise('/');
    })
```

10 Оновіть DishDetailController для використання $stateParams як наведено:

```
        .controller('DishDetailController', ['$scope', '$stateParams', 'menuFactory', function($scope, $stateParams, menuFactory) {
            var dish= menuFactory.getDish(parseInt($stateParams.id,10));
                        $scope.dish = dish;
                    }])
```

11 Створіть представлення у каталозі View: У каталозі app створіть каталог views і перемістіть всі представлення в цей каталог. В цьому ж каталозі створіть два нові файли – header.html і footer.html. З файлу index.html виріжіть частини <nav> і <header> і перемістіть їх в header.html. Аналогічно з файлу index.htm перемістіть код з тегу <footer> до файлу footer.html. В index.html, замініть ngView на uiView:

```
    <div ui-view="header"></div>
    <div ui-view="content"></div>
    <div ui-view="footer"></div>
```

Оновіть тег <scripts> для використання angular-ui-router.min.js замість angular-route.js:

```
    <script src="../bower_components/angular-ui-router/release/angular-ui-router.min.js"></script>
```

Відкрийте header.html і оновіть hrefs використовуючи ui-srefs як показано нижче:

```
                <a class="navbar-brand" ui-sref="app"><img src="images/logo.png" height=30 width=41></a>
            </div>
            <div id="navbar" class="navbar-collapse collapse">
                <ul class="nav navbar-nav">
                    <li><a ui-sref="app">
                        <span class="glyphicon glyphicon-home"
```

```
                        aria-hidden="true"></span>
Home</a></li>
                    <li><a ui-sref="app.aboutus">
                        <span  class="glyphicon  glyphicon-info-
sign"
                        aria-hidden="true"></span>
About</a></li>
                    <li><a ui-sref="app.menu">
                        <span  class="glyphicon  glyphicon-list-
alt"
                        aria-hidden="true"></span>
                         Menu</a></li>
                     <li><a ui-sref="app.contactus">
                     <i       class="fa       fa-envelope-o"></i>
Contact</a></li>
                </ul>
            </div>
```

Оновіть hrefs в footer з використанням ui-srefs як наведено:

```
                    <ul class="list-unstyled">
                        <li><a ui-sref="app">Home</a></li>
                        <li><a                                  ui-
sref="app.aboutus">About</a></li>
                        <li><a ui-sref="app.menu">Menu</a></li>
                        <li><a                                  ui-
sref="app.contactus">Contact</a></li>
                    </ul>
```

Переконайтесь що перемістили menu.html, contactus.html і dishdetail.html до каталогу views.

Відредагуйте menu.html шляхом заміни href на ui-sref:

```
                        <a          ui-sref="app.dishdetails({id:
dish._id})">
                        <img class="media-object img-thumbnail"
                        ng-src={{dish.image}} alt="Uthappizza">
                        </a>
```

Відредагуйте файл dishdetail.html так, щоб включити кнопку для відображення меню:

```
            <div class="col-xs-12">
               <button   class="btn   btn-xs   btn-primary   pull-
right"
                    type="button" ui-sref="app.menu">
                    Back to Menu
                </button>
              <div class="media">
```

Збережіть всі файли в перегляньте завершений SPA.

4.3 Зміст звіту

Звіт має містити:

- титульний аркуш;
- мету роботи і завдання;

– покроковий опис роботи, програмний код для реалізації кожного завдання, копії екранів з результатами виконаної роботи.
– висновки.

Запитання для самоконтролю:

Тривалість заняття: 6 год.

ЛАБОРАТОРНА РОБОТА 5.
МОДУЛЬНЕ ТЕСТУВАННЯ ANGULAR ДОДАТКІВ

Мета роботи: одержати навики у застосуванні модульного тестування в Angular.

Обладнання:

– доступ до мережі internet;

– ПК IBM PC x86 CPU з встановленою операційною системою.

5.1 Теоретичні відомості

Налаштування середовища для модульного тестування:

1 Встановлення доступу до ядра Jasmine з поточного проекту:

```
npm install jasmine-core --save-dev
```

2 Налаштування інструментів командного рядка Karma глобально наступним чином:

```
npm install karma-cli -g
```

Використовуйте sudo в операційній системі Linux.

3 Встановіть пагін karma-jasmine для можливості використання Jasmine з Karma:

```
npm install karma-jasmine --save-dev
```

4 Для того, щоб налаштувати середовища браузера для проведення тестувань, встановіть PhantomJS і Karma завантажувачі для PhantomJS і Chrome наступним чином:

```
npm install phantomjs karma-phantomjs-launcher karma-chrome-launcher --save-dev
```

Можна також встановити для браузерів Firefox, IE і Safari.

Налаштування Angular Mocks:

5 Необхідно становити модуль ngMock:

```
bower install angular-mocks -S
```

Модульне тестування MenuController

6 Налаштуємо Karma для проведення модульного тестування. Створимо каталог *test* в каталозі *conFusion*.

7 Перейдіть в каталог *test* і створіть там файл з назвою *karma.conf.js*. Цей файл буде містити конфігураційну інформацію для тестів Karma. Додайте наступне в цей файл:

```
// Karma configuration

module.exports = function(config) {
  config.set({

    // base path that will be used to resolve all patterns (eg.
files, exclude)
    basePath: '../',

    // frameworks to use
    //                    available                    frameworks:
https://npmjs.org/browse/keyword/karma-adapter
    frameworks: ['jasmine'],

  })
}
```

8 Додайте список файлів, які будуть включені, і ті, які будуть виключені в конфігурації наступним чином:

```
    // list of files / patterns to load in the browser
    files: [
      'bower_components/angular/angular.js',
      'bower_components/angular-resource/angular-resource.js',
      'bower_components/angular-ui-router/release/angular-ui-
router.js',
      'bower_components/angular-mocks/angular-mocks.js',
      'app/scripts/*.js',
      'test/unit/**/*.js'
    ],

    // list of files to exclude
    exclude: [
    'test/protractor.conf.js', 'test/e2e/*.js'
    ],
```

9 Додайте деяку інформацію про конфігурацію в такий спосіб (детальніше дивіться коментарі):

```
    // preprocess matching files before serving them to the
browser
    //                  available                  preprocessors:
https://npmjs.org/browse/keyword/karma-preprocessor
    preprocessors: {
    },

    // test results reporter to use
    // possible values: 'dots', 'progress'
    //                    available                    reporters:
https://npmjs.org/browse/keyword/karma-reporter
    reporters: ['progress'],

    // web server port
    port: 9876,

    // enable / disable colors in the output (reporters and
logs)
    colors: true,
```

```
    // level of logging
    // possible values: config.LOG_DISABLE || config.LOG_ERROR
|| config.LOG_WARN || config.LOG_INFO || config.LOG_DEBUG
    logLevel: config.LOG_INFO,

    // enable / disable watching file and executing tests
whenever any file changes
    autoWatch: true,
```

10 Далі, необхідно налаштовувати браузери для використання при тестуванні в такий спосіб:

```
    // start these browsers
    //           available           browser           launchers:
https://npmjs.org/browse/keyword/karma-launcher
     browsers: ['Chrome','PhantomJS', 'PhantomJS_custom'],

    // you can define custom flags
    customLaunchers: {
      'PhantomJS_custom': {
        base: 'PhantomJS',
        options: {
          windowName: 'my-window',
          settings: {
            webSecurityEnabled: false
          },
        },
        flags: ['--load-images=true'],
        debug: true
      }
    },

    phantomjsLauncher: {
      // Have phantomjs exit if a ResourceError is encountered
(useful if karma exits without killing phantom)
      exitOnResourceError: true
    },

    // Continuous Integration mode
    // if true, Karma captures browsers, runs the tests and
exits
    singleRun: false,

    // Concurrency level
    // how many browser should be started simultaneous
    concurrency: Infinity
```

11 Збережіть зміни. Тепер конфігурація Karma готова до запуску тесту. Далі створюємо тест для MenuController.

12 Створіть каталог з назвою *unit* в каталозі *test.* Перемістіться в каталог *unit* і там створіть каталог з ім'ям *controllers*. Потім перейдіть в каталог *controllers*. В цьому каталозі будуть зберігатись тести для контролерів.

13 Створіть файл з назвою *menucontroller.js і* почніть конфігурацію тесту наступним чином:

```
describe('Controller: MenuController', function () {

  // load the controller's module
  beforeEach(module('confusionApp'));

  var MenuController, scope, $httpBackend;

});
```

14 Тепер будемо вводити макетні залежності наступним чином:

```
  // Initialize the controller and a mock scope
  beforeEach(inject(function    ($controller,    _$httpBackend_,
$rootScope, menuFactory) {

          // place here mocked dependencies
      $httpBackend = _$httpBackend_;


$httpBackend.expectGET("http://localhost:3000/dishes").respond([
        {
      "id": 0,
      "name": "Uthapizza",
      "image": "images/uthapizza.png",
      "category": "mains",
      "label": "Hot",
      "price": "4.99",
      "description": "A",
      "comments":[{}]
      },
      {
      "id": 1,
      "name": "Zucchipakoda",
      "image": "images/zucchipakoda.png",
      "category": "mains",
      "label": "New",
      "price": "4.99",
      "description": "A",
      "comments":[{}]
      }
      ]);

    scope = $rootScope.$new();
    MenuController = $controller('MenuController', {
      $scope: scope, menuFactory: menuFactory
    });
            $httpBackend.flush();

  }));
```

15 Налаштуйте всі тести в файлі наступним чином:

```
  it('should have showDetails as false', function () {

    expect(scope.showDetails).toBeFalsy();

  });
```

```
  it('should create "dishes" with 2 dishes fetched from xhr',
function(){

      expect(scope.showMenu).toBeTruthy();
      expect(scope.dishes).toBeDefined();
      expect(scope.dishes.length).toBe(2);

  });

  it('should have the correct data order in the dishes',
function() {

      expect(scope.dishes[0].name).toBe("Uthapizza");
      expect(scope.dishes[1].label).toBe("New");

  });

  it('should change the tab selected based on tab clicked',
function(){

      expect(scope.tab).toEqual(1);

      scope.select(3);

      expect(scope.tab).toEqual(3);
      expect(scope.filtText).toEqual('mains');

  });
```

16 Збережіть зміни. Тепер тест готовий до виконання.

17 Поверніться в каталог *test*, а потім введіть наступну команду в командному рядку для виконання тесту:

```
karma start karma.conf.js
```

18 Всі тести повинні успішно завершитись. Спробуйте змінити деякі із значень тестів, щоб викликати випадок непроходження тесту.

5.2 Порядок виконання роботи

1 Виконайте пункти 1-18 лабораторної роботи.
2 Налаштуйте конфігураційний файл Karma.
3 Напишіть модульні тести з використанням Jasmine.
4 Дослідіть виконання тестів шляхом зміни їх параметрів.
5 Оформити звіт.

5.3 Зміст звіту

Звіт має містити:

- титульний аркуш;
- мету роботи і завдання;

– покроковий опис роботи, програмний код для реалізації кожного завдання, копії екранів з результатами виконаної роботи.
– висновки.

Запитання для самоконтролю:

Тривалість заняття: 4 год.

ЛАБОРАТОРНА РОБОТА 6.
E2E ТЕСТУВАННЯ ANGULAR ДОДАТКІВ

Мета роботи: одержати навики у застосуванні системного тестування (e2e) в Angular; одержати навики у використанні Protractor при проведенні e2e тестування.

Обладнання:

– доступ до мережі internet;

– ПК IBM PC x86 CPU з встановленою операційною системою.

6.1 Теоретичні відомості

Налаштування середовища тестування EndToEnd (e2e):

1 Встановіть інструмент protractor глобально для використання в тестуванні e2e:

```
npm install protractor -g
```

Використовуйте sudo при роботі в операційній системі Linux.

2 Вище вказаний інструмент також встановлює webdriver-manager. Потім необхідно оновити web-драйвери, набравши:

```
webdriver-manager update
```

Використовуйте sudo при роботі в операційній системі Linux.

3 Переконайтеся, що ви запускаєте JSON-сервер, щоб обслуговувати REST API для доступу до даних в форматі JSON за допомогою додатку Angular.

4 Далі, необхідно запустити сервер, щоб обслуговувати додаток Angular. Gulp вже налаштований для виконання такої задачі. Переконайтеся, що поточний каталог – це *conFusion*. У командному рядку введіть команду:

```
gulp watch
```

Ця команда запустить сервер і буде обслуговувати сторінку за адресою: http://localhost:3001/.

Налаштування Protractor:

5 Перейдіть в каталог test і створіть файл з назвою *protractor.conf.js* та помістіть в нього наступний конфігураційний код:

```
exports.config = {
```

```
  allScriptsTimeout: 11000,
  specs: [
    'e2e/*.js'
  ],
  capabilities: {
    'browserName': 'chrome'
  },

  baseUrl: 'http://localhost:3001/',

  framework: 'jasmine',
    directConnect: true,

  jasmineNodeOpts: {
    defaultTimeoutInterval: 30000
  }
};
```

Тут ми використовуємо опцію DirectConnect для прямого підключення до браузера для проведення тестувань, а не через web-сервер Selenium.

6 Створіть каталог з назвою e2e в каталозі test і перейдіть в щойно створений каталог.

7 Створіть файл з назвою *scenarios.js*. Він буде містити всі e2e тести, що будуть проведені з додатком.

8 Додайте наступний код до файлу:

```
'use strict';

describe('conFusion App E2E Testing', function() {

});
```

9 Далі необхідно ввести тести в цей файл. Спочатку переконайтесь, що відбувається перенаправлення в файл index.html:

```
  it('should automatically redirect to / when location
hash/fragment is empty', function() {

    browser.get('index.html');
    expect(browser.getLocationAbsUrl()).toMatch("/");

  });
```

10 Створення простого тесту для файлу index для перевірки правильності встановлення заголовку сторінки:

```
describe('index', function() {
    beforeEach(function() {
      browser.get('index.html#/');
    });

    it('should have a title', function() {
      expect(browser.getTitle()).
        toEqual('Ristorante Con Fusion');
```

```
    });
  });
```

11 Тестування кількох властивостей першого пункту меню:

```
describe('menu 0 item', function() {
    beforeEach(function() {
      browser.get('index.html#/menu/0');
    });

    it('should have a name', function() {
          var name = element(by.binding('dish.name'));
          expect(name.getText()).
             toEqual('Uthapizza Hot $4.99');
    });

    it('should show the number of comments as', function() {
         expect(element.all(by.repeater('comment                in
dish.comments'))
             .count()).toEqual(5);

    });

    it('should show the first comment author as', function() {
          element(by.model('orderText')).sendKeys('author');
            expect(element.all(by.repeater('comment             in
dish.comments'))
             .count()).toEqual(5);
          var   author   =   element.all(by.repeater('comment   in
dish.comments'))

.first().element(by.binding('comment.author'));

          expect(author.getText()).toContain('25 Cent');

    });
 });
```

12 Збережіть зміни і перейдіть назад в каталог test.

13 У командному рядку введіть наступну команду для виконання тестів e2e:

```
    protractor protractor.conf.js
```

14 Всі тести повинні пройти успішно. Змініть деякі з вхідних даних тесту, щоб викликати випадок непроходження тесту.

6.2 Порядок виконання роботи

1 Виконати пункти 1-14 лабораторної роботи.
2 Написати системні тести.
3 Дослідити виконання тестів шляхом зміни їх параметрів.
4 Оформити звіт.

6.3 Зміст звіту

Звіт має містити:

- титульний аркуш;
- мету роботи і завдання;
- покроковий опис роботи, програмний код для реалізації кожного завдання, копії екранів з результатами виконаної роботи.
- висновки.

Запитання для самоконтролю:

Тривалість заняття: 4 год.

ЛАБОРАТОРНА РОБОТА 7.
НАЛАШТУВАННЯ ПРОСТОГО СЕРВЕРА З ВИКОРИСТАННЯМ МОДУЛЯ JSON-SERVER

Мета роботи: одержати навики у налаштуванні і запуску сервера за допомогою *json-server* для обслуговування даних Angular додатку.

Обладнання:

– доступ до мережі internet;

– ПК IBM PC x86 CPU з встановленою операційною системою.

7.1 Теоретичні відомості

Модуль Node *json-server* забезпечує простий спосіб налаштувати web-сервер, який підтримує повноцінний REST API сервер. Він також може обслуговувати статичний контент з каталогу. В роботі буде використано вказані дві функції для забезпечення back-end для додатку Angular.

Встановлення json-server:

1 *json-server* – модуль Node, встановлюється глобально за допомогою команди:

```
npm install json-server -g
```

За умови використання Linux застосуйте s**udo.** Це дозволить встановити *json-server*, який може бути запущений з командного рядка з будь-якого каталогу на комп'ютері.

Налаштування каталогу сервера:

2 В будь-якому зручному місці на вашому комп'ютері створіть новий каталог з назою **json-server** і перейдіть до цього каталогу.

3 Збережіть код, наведений нижче в файл з назвою db.json і помістіть вказаний файл в щойно створений каталог.

```
{
  "dishes": [
    {
      "id": 0,
      "name": "Uthapizza",
      "image": "images/uthapizza.png",
      "category": "mains",
      "label": "Hot",
```

```
      "price": "4.99",
      "description": "A unique combination of Indian Uthappam (pancake) and Italian pizza, topped with Cerignola olives, ripe vine cherry tomatoes, Vidalia onion, Guntur chillies and Buffalo Paneer.",
      "comments": [
        {
          "rating": 5,
          "comment": "Imagine all the eatables, living in conFusion!",
          "author": "John Lemon",
          "date": "2012-10-16T17:57:28.556094Z"
        },
        {
          "rating": 4,
          "comment": "Sends anyone to heaven, I wish I could get my mother-in-law to eat it!",
          "author": "Paul McVites",
          "date": "2014-09-05T17:57:28.556094Z"
        },
        {
          "rating": 3,
          "comment": "Eat it, just eat it!",
          "author": "Michael Jaikishan",
          "date": "2015-02-13T17:57:28.556094Z"
        },
        {
          "rating": 4,
          "comment": "Ultimate, Reaching for the stars!",
          "author": "Ringo Starry",
          "date": "2013-12-02T17:57:28.556094Z"
        },
        {
          "rating": 2,
          "comment": "It's your birthday, we're gonna party!",
          "author": "25 Cent",
          "date": "2011-12-02T17:57:28.556094Z"
        }
      ]
    },
    {
      "id": 1,
      "name": "Zucchipakoda",
      "image": "images/zucchipakoda.png",
      "category": "appetizer",
      "label": "",
      "price": "1.99",
      "description": "Deep fried Zucchini coated with mildly spiced Chickpea flour batter accompanied with a sweet-tangy tamarind sauce",
      "comments": [
        {
          "rating": 5,
          "comment": "Imagine all the eatables, living in conFusion!",
          "author": "John Lemon",
          "date": "2012-10-16T17:57:28.556094Z"
        },
        {
          "rating": 4,
```

```
          "comment": "Sends anyone to heaven, I wish I could get
my mother-in-law to eat it!",
          "author": "Paul McVites",
          "date": "2014-09-05T17:57:28.556094Z"
        },
        {
          "rating": 3,
          "comment": "Eat it, just eat it!",
          "author": "Michael Jaikishan",
          "date": "2015-02-13T17:57:28.556094Z"
        },
        {
          "rating": 4,
          "comment": "Ultimate, Reaching for the stars!",
          "author": "Ringo Starry",
          "date": "2013-12-02T17:57:28.556094Z"
        },
        {
          "rating": 2,
          "comment": "It's your birthday, we're gonna party!",
          "author": "25 Cent",
          "date": "2011-12-02T17:57:28.556094Z"
        }
      ]
    },
    {
      "id": 2,
      "name": "Vadonut",
      "image": "images/vadonut.png",
      "category": "appetizer",
      "label": "New",
      "price": "1.99",
      "description": "A quintessential ConFusion experience, is
it a vada or is it a donut?",
      "comments": [
        {
          "rating": 5,
          "comment": "Imagine all the eatables, living in
conFusion!",
          "author": "John Lemon",
          "date": "2012-10-16T17:57:28.556094Z"
        },
        {
          "rating": 4,
          "comment": "Sends anyone to heaven, I wish I could get
my mother-in-law to eat it!",
          "author": "Paul McVites",
          "date": "2014-09-05T17:57:28.556094Z"
        },
        {
          "rating": 3,
          "comment": "Eat it, just eat it!",
          "author": "Michael Jaikishan",
          "date": "2015-02-13T17:57:28.556094Z"
        },
        {
          "rating": 4,
          "comment": "Ultimate, Reaching for the stars!",
          "author": "Ringo Starry",
          "date": "2013-12-02T17:57:28.556094Z"
```

```
        },
        {
          "rating": 2,
          "comment": "It's your birthday, we're gonna party!",
          "author": "25 Cent",
          "date": "2011-12-02T17:57:28.556094Z"
        }
      ]
    },
    {
      "id": 3,
      "name": "ElaiCheese Cake",
      "image": "images/elaicheesecake.png",
      "category": "dessert",
      "label": "",
      "price": "2.99",
      "description": "A delectable, semi-sweet New York Style Cheese Cake, with Graham cracker crust and spiced with Indian cardamoms",
      "comments": [
        {
          "rating": 5,
          "comment": "Imagine all the eatables, living in conFusion!",
          "author": "John Lemon",
          "date": "2012-10-16T17:57:28.556094Z"
        },
        {
          "rating": 4,
          "comment": "Sends anyone to heaven, I wish I could get my mother-in-law to eat it!",
          "author": "Paul McVites",
          "date": "2014-09-05T17:57:28.556094Z"
        },
        {
          "rating": 3,
          "comment": "Eat it, just eat it!",
          "author": "Michael Jaikishan",
          "date": "2015-02-13T17:57:28.556094Z"
        },
        {
          "rating": 4,
          "comment": "Ultimate, Reaching for the stars!",
          "author": "Ringo Starry",
          "date": "2013-12-02T17:57:28.556094Z"
        },
        {
          "rating": 2,
          "comment": "It's your birthday, we're gonna party!",
          "author": "25 Cent",
          "date": "2011-12-02T17:57:28.556094Z"
        }
      ]
    }
  ],
  "promotions": [
    {
      "id": 0,
      "name": "Weekend Grand Buffet",
      "image": "images/buffet.png",
```

```
      "label": "New",
      "price": "19.99",
      "description": "Featuring mouthwatering combinations with a choice of five different salads, six enticing appetizers, six main entrees and five choicest desserts. Free flowing bubbly and soft drinks. All for just $19.99 per person "
    }
  ],
  "leadership": [
    {
      "id": 0,
      "name": "Peter Pan",
      "image": "images/alberto.png",
      "designation": "Chief Epicurious Officer",
      "abbr": "CEO",
      "description": "Our CEO, Peter, credits his hardworking East Asian immigrant parents who undertook the arduous journey to the shores of America with the intention of giving their children the best future. His mother's wizardy in the kitchen whipping up the tastiest dishes with whatever is available inexpensively at the supermarket, was his first inspiration to create the fusion cuisines for which The Frying Pan became well known. He brings his zeal for fusion cuisines to this restaurant, pioneering cross-cultural culinary connections."
    },
    {
      "id": 1,
      "name": "Dhanasekaran Witherspoon",
      "image": "images/alberto.png",
      "designation": "Chief Food Officer",
      "abbr": "CFO",
      "description": "Our CFO, Danny, as he is affectionately referred to by his colleagues, comes from a long established family tradition in farming and produce. His experiences growing up on a farm in the Australian outback gave him great appreciation for varieties of food sources. As he puts it in his own words, Everything that runs, wins, and everything that stays, pays!"
    },
    {
      "id": 2,
      "name": "Agumbe Tang",
      "image": "images/alberto.png",
      "designation": "Chief Taste Officer",
      "abbr": "CTO",
      "description": "Blessed with the most discerning gustatory sense, Agumbe, our CFO, personally ensures that every dish that we serve meets his exacting tastes. Our chefs dread the tongue lashing that ensues if their dish does not meet his exacting standards. He lives by his motto, You click only if you survive my lick."
    },
    {
      "id": 3,
      "name": "Alberto Somayya",
      "image": "images/alberto.png",
      "designation": "Executive Chef",
      "abbr": "EC",
      "description": "Award winning three-star Michelin chef with wide International experience having worked closely with
```

```
whos-who in the culinary world, he specializes in creating
mouthwatering Indo-Italian fusion experiences. He says, Put
together the cuisines from the two craziest cultures, and you
get a winning hit! Amma Mia!"
    }
  ],
  "feedback": [

  ]
}
```

4 Змініть поточний каталог у вікні терміналу на каталог **json-server** і введіть наступну команду в командному рядку для запуску сервера:

```
    json-server --watch db.json
```

5 Виконана дія повинна запустити сервер на порту з номером 3000 на вашій машині. Дані з цього сервера можна отримати, ввівши наступні адреси в адресний рядок браузера:

```
http://localhost:3000/dishes
http://localhost:3000/promotions
http://localhost:3000/leadership
http://localhost:3000/feedback
```

6 Введіть ці адреси в браузер для відображення даних в форматі JSON, які обслуговуються сервером. Ці дані отримані з файлу db.json.

7 Json-server також є статичним web-сервером. Будь-які ресурси, які поміщені в каталог з назвою **public** в каталозі **json-server**, будуть обслуговуватися сервером за наступною адресою:

```
  http://localhost:3000/
```

8 Зупиніть сервер командою **ctrl-C** в терміналі.

Налаштування файлу gulpfile.js для створення каталогу Dist:

9 Файл gulpfile.js раніше був налаштований для генерування каталогу dist з конфігурації, представленої в файлі menu.html. Тепер необхідно оновити gulpfile.js для використання файлу index.html для конфігувації. Оновіть завдання usemin у файлі gulpfile.js як показано нижче:

```
gulp.task('usemin',['jshint'], function () {
  return gulp.src('./app/**/*.html')
      .pipe(usemin({
        css:[minifycss(),rev()],
        js: [ngannotate(),uglify(),rev()]
      }))
      .pipe(gulp.dest('dist/'));
});
```

10 Для завдання browser-sync оновіть налаштування, як показано нижче:

```
browserSync.init(files, {
   server: {
      baseDir: "dist",
      index: "index.html"
   }
});
```

11 Тепер при запуску gulp в командному рядку, він створює каталог dist, яка містить файли дистрибутива.

12 Скопіюйте увесь вміст каталогу dist в каталог public, яка була створена вище.

13 Перезавантажте сервер. Тепер можна отримати доступ до web-сайту, який обслуговується сервером, набравши http://localhost:3000/ в адресному рядку браузера.

7.2 Порядок виконання роботи

1 Виконати пункти 1-13 лабораторної роботи.

2 Налаштувати і запустити простий сервер з використанням модуля **json-server**.

3 Продемонструвати можливість обслуговування статичного контенту сервером на прикладі власної web-сторінки.

4 Оформити звіт.

7.3 Зміст звіту

Звіт має містити:

- титульний аркуш;
- мету роботи і завдання;
- покроковий опис роботи, програмний код для реалізації кожного завдання, копії екранів з результатами виконаної роботи.
- висновки.

Запитання для самоконтролю:

Тривалість заняття: 4 год.

ЛАБОРАТОРНА РОБОТА 8.
ОБРОБКА ПОМИЛОК У КЛІЄНТ-СЕРВЕРНІЙ ВЗАЄМОДІЇ

Мета роботи: одержати навики у налаштуванні контролерів для запровадження можливості обробки помилок при клієнт-серверній взаємодії.

Обладнання:

– доступ до мережі internet;

– ПК IBM PC x86 CPU з встановленою операційною системою.

8.1 Теоретичні відомості

Обробка помилок при клієнт-серверній взаємодії з використанням $http

Оновлення контролерів:

1 Відкрийте *controllers.js* і оновіть код для обробки помилок. Оновлення коду в *MenuController*:

```
            $scope.showMenu = false;
            $scope.message = "Loading ...";
                        $scope.dishes= {};
                        menuFactory.getDishes()
            .then(
                function(response) {
                    $scope.dishes = response.data;
                    $scope.showMenu = true;
                },
                function(response) {
                    $scope.message = "Error: "+response.status +
" " + response.statusText;
                }
            );
```

2 Далі відкрийте menu.html і оновіть код наступним чином:

```
        <div          class="row          row-content"          ng-
controller="MenuController">
            <div class="col-xs-12" ng-if="!showMenu">
                <h3>{{message}}</h3>
            </div>
            <div class="col-xs-12" ng-if="showMenu">
```

Зверніть увагу на використання директиви ngIf для того, щоб додати / видалити div з DOM.

3 Потім, оновіть *DishDetailController* в *controllers.js* наступним чином:

```
            $scope.dish = {};
```

```
            $scope.showDish = false;
            $scope.message="Loading ...";

menuFactory.getDish(parseInt($stateParams.id,10))
            .then(
                function(response){
                    $scope.dish = response.data;
                    $scope.showDish=true;
                },
                function(response) {
                    $scope.message = "Error: "+response.status +
" " + response.statusText;
                }
            );
```

4 Також оновіть *IndexController:*

```
                        $scope.dish = {};
                        $scope.showDish = false;
                        $scope.message="Loading ...";

                        menuFactory.getDish(0)
                        .then(
                            function(response){
                                $scope.dish = response.data;
                                $scope.showDish = true;
                            },
                            function(response) {
                                $scope.message     =     "Error:
"+response.status + " " + response.statusText;
                            }
                        );
```

5 Оновіть вміст файлу *dishdetail.html:*

```
        <div          class="row          row-content"          ng-
controller="DishDetailController">
            <div class="col-xs-12" ng-if="!showDish">
                <h3>{{message}}</h3>
            </div>
            <div class="col-xs-12" ng-if="showDish">
```

6 Оновіть вміст файлу *home.html:*

```
            <div class="col-xs-12 col-sm-9 col-sm-pull-3">
                <div ng-if="!showDish">
                    <h3>{{message}}</h3>
                </div>
                <div class="media" ng-if="showDish">
```

7 Збережіть всі зміни і перегляньте одержану web-сторінку.

Обробка помилок при клієнт-серверній взаємодії з використанням $resource

Оновлення контролерів:

8 Відкрийте *controllers.js* і оновіть код в *MenuController*:

```
            $scope.showMenu = false;
            $scope.message = "Loading ...";
                        menuFactory.getDishes().query(
```

```
                function(response) {
                    $scope.dishes = response;
                    $scope.showMenu = true;
                },
                function(response) {
                    $scope.message = "Error: "+response.status +
" " + response.statusText;
                });
```

9 Аналогічно оновіть *DishDetailController:*

```
            $scope.showDish = false;
            $scope.message="Loading ...";
                        $scope.dish                                 =
menuFactory.getDishes().get({id:parseInt($stateParams.id,10)})
            .$promise.then(
                            function(response){
                                $scope.dish = response;
                                $scope.showDish = true;
                            },
                            function(response) {
                                $scope.message     =     "Error:
"+response.status + " " + response.statusText;
                            }
            );
```

10 Оновіть *IndexController:*

```
                        $scope.showDish = false;
                        $scope.message="Loading ...";
                        $scope.dish                                 =
menuFactory.getDishes().get({id:0})
                        .$promise.then(
                            function(response){
                                $scope.dish = response;
                                $scope.showDish = true;
                            },
                            function(response) {
                                $scope.message     =     "Error:
"+response.status + " " + response.statusText;
                            }
                        );
```

Оновлення інформації на сервері:

11 Тепер ви будете оновлювати *DishCommentController* для того, щоб забезпечити передавання зауважень користувача про страву до сервера. Введіть *menuFactory* до *DishCommentController*:

```
        .controller('DishCommentController',          ['$scope',
'menuFactory', function($scope,menuFactory) {
```

12 Оновіть функцію *submitComment()*:

```
            $scope.submitComment = function () {
                                $scope.mycomment.date    =    new
Date().toISOString();
                console.log($scope.mycomment);

$scope.dish.comments.push($scope.mycomment);
```

```
menuFactory.getDishes().update({id:$scope.dish.id},$scope.dish);

$scope.commentForm.$setPristine();
                                    $scope.mycomment   =   {rating:5,
comment:"", author:"", date:""};
            }
```

13 Збережіть зміни і перегляньте результат виконання оновленої web-сторінки.

8.2 Порядок виконання роботи

1 Виконайте пункти 1-7 лабораторної роботи.

2 Додайте можливість обробки помилок з використанням **$http до своєї web-сторінки.**

3 Виконайте пункти 8-13 лабораторної роботи.

4 Додайте можливість обробки помилок з використанням **$resource до своєї web-сторінки**.

5 Заберпечте можливість оновлення даних на сервері.

6 Оформити звіт.

8.3 Зміст звіту

Звіт має містити:

– титульний аркуш;

– мету роботи і завдання;

– покроковий опис роботи, програмний код для реалізації кожного завдання, копії екранів з результатами виконаної роботи.

– висновки.

Запитання для самоконтролю:

Тривалість заняття: 4 год.

ПЕРЕЛІК РЕКОМЕНДОВАНИХ ДЖЕРЕЛ

1 Chandermani AngularJS by Example [Електронний ресурс] / Chandermani. – Packt Publishing. – 2015. – 454 p. – Режим доступу: http://it-ebooks.org/book/packt/angularjs_by_example

2 Frisbie M. AngularJS Web Application Development Cookbook [Електронний ресурс] / Matt Frisbie. - Packt Publishing. - 2014. - 346 p. - Режим доступу: http://it-ebooks.org/book/packt/angularjs_web_application_development_cookbook

3 Green B. Angular JS: Up and Running [Електронний ресурс] / Brad Green, Shyam Seshadri. - Published by O'Reilly Media, Inc. - 2014. - 302 p. - Режим доступу: http://it-ebooks.org/book/oreilly/angularjs_up_and_running

4 Безгачнюк Ю. В. Сучасні технології програмування: конспект лекцій / Ю. В. Безгачнюк, Л. О. Штаєр. – Івано-Франківськ: ІФНТУНГ, 2011. – 161 с.

5 Безгачнюк Ю. В. Сучасні технології програмування: лабораторний практикум. – Івано-Франківськ: ІФНТУНГ, 2011. – 80 с.

6 Гудман Д. JavaScript.: Библия пользователя. – 4-е изд. – М.: Издательский дом "Вильямс", 2003. – 960 с.

7 Резиг Дж. JavaScript для профессионалов / Резиг Дж., Фергюсон Р., Пакстон Дж. – М: ООО «И.Д. Вильямс», 2016. – 240 с. - Режим доступу: http://owlweb.ru/javascript-dlya-professionalov-2-e-izdanie-rezig-dzh-fergyuson-r-pakston-dzh/

8 Тимків Д.Ф. Архітектура та проектування програмного забезпечення: конспект лекцій / Д.Ф.Тимків, В.М.Юрчишин, В.І.Шекета. - Івано-Франківськ: ІФНТУНГ, 2013. - 53 с.