با توجه به مفاهیم و فناوری هایی که در طول کلاس درس برنامه نویسی وب آموختید باید در این پروژه به پیاده سازی قسمت سمت سرور سایت سفارش غذا بپردازید.

در این قسمت پروژه شما باید کد مربوط به سرور و دیتابیس سایت سفارش غذا را پیاده سازی کنید. کار شما با طراحی و پیاده سازی دیتابیس مورد استفاده توسط پروژه شروع میشود. شما سپس می‌بایست برنامه سمت سرور خود را بنویسید آن را به پایگاه داده وصل کرده و سپس داده ها را به طریقی در اختیار کد سمت کاربر قرار دهید. توجه داشته باشید که پیاده سازی سمت سرور فقط با استفاده از زبان های گفته شده و یا فریم ورک های آنها مجاز است.

باید به این نکته توجه داشته باشید که پیاده سازی قسمت سمت سرور باید در قالب یک API باشد. به این معنی که در نهایت باید سرویسهای پروژه را به شکل مجموعه‌ای از endpointها به کاربر ارائه کنید. برای مثال فرض کنید میخواهید سرویسی پیاده سازی کنید که لیست رستورانها را به شما برمیگرداند. شما در زبان انتخابی خود کدی میزنید و سرور خود را روی localhost اجرا میکنید. حالا باید بتوانید با یک درخواست HTTP از نوع GET به یک url خاص (مثل: restaurants/localhost) لیستی از رستوران ها را به صورت JSON دریافت کنید. تمامی سرویس ها باید در چنین قالبی باشند.

هیچ محدودیتی در انتخاب زبان، تکنولوژی و Framework برای این بخش وجود ندارد و استفاده از یکی از گزینه های زیر پیشنهاد میشود:

Language	Frameworks	Document
PHP	Laravel - Lumen - [pure php]	https://laravel.com/docs https://lumen.laravel.com/docs
(javascript) node.js	Express	https://expressjs.com http://yon.ir/sBYhN
Python	Flask - Django	http://flask.pocoo.org http://yon.ir/9WW9o

طراحی پایگاه داده

شما میتوانید از هر تکنولوژی برای پیاده‌سازی پایگاه داده خود استفاده کنید (توصیه می‌شود از mongodb استفاده کنید)، تنها نکته‌ای که ملزم به رعایت آن هستید پیروی از ساختار دادههای توصیف شده است.

در این پروژه چندین نوع داده وجود دارد که هرکدام میتوانند در قالب یک جدول یا Schema پیاده‌سازی شوند. این جداول عبارتند از :

- نظرات کاربران
- رستورانها
- غذاها
- آدرسها
- دسته بندیها

در ادامه به توصیف کلی هر یک ازین ساختارها (موجودیت یا جدول) می‌پردازیم:

نظرات کاربران

هر کاربر می تواند نظرات خود را در رابطه با رستوران اعلام کند. هر نظری که کاربر ثبت می کند مربوط به یک رستوران است و هر رستوران می تواند چندین نظر داشته باشد.(راهنمایی : رابطه بین موجودیت نظرات و رستوران ها یک به چند است). ساختار کلی شمای نظرات به صورت زیر است :

```
Comment = new schema({
  id: String, // or an auto increment number,
  author: String,
  // rates
  quality: Number, // a number between 0-5
  packaging: Number,
  deliveryTime: Number,
  text: String,
  created_at: Date, // time where comment submitted
});
```

رستوران‌ها

هر رستوران ویژگی های زیر را داراست. کاربرد فیلدهای openingTimeو closingTime برای محاسبه ساعات باز و بسته بودن رستوران می‌باشد. دقت داشته باشید که فرض بر این است که رستوران ها همه روزه فعال هستند. فیلد address نشان دهنده ارتباط رستوران با یک آدرس است که در قسمت های بعدی به توضیح آن می پردازیم. فیلد foodSets نشان دهنده مجموعه غذاهایی است که هر رستوران دارد و ارائه می دهد.

```
Restaurant = new schema({
  id: String, // or an autoincreament number,
```

```
  name:String,
  logo:String, // src of logo image
  openingTime:Number, // time of opening
  closingTime:Number, // time of closing
  averageRate:Number, // average of comments rate
  address: AddressSchema,
  categories:[CategorySchema], // array of food categories. e.g. fastfood or irani
  foods:[FoodSchema],
  comments:[CommentSchema],
})
```

غذاها

هر غذا متعلق به یک رستوران است و هر رستوران چندین نوع غذا دارد. دقت داشته باشید که با توجه به اینکه قیمت و مشخصات هر غذا در هر رستوران ممکن است متفاوت از یک رستوران دیگری باشد (برای مثال قرمه سبزی در یک رستوران ۱۵ هزارتومان و در دیگری ۲۰ هزارتومان همراه با یک سری تفاوت در توضیحات غذا) و برای سادگی پیاده سازی پایگاه داده هر رستوران غذاهای مختص خود را دارد. یعنی ممکن است غذای x در جدول مربوط به غذاها 100 بار آمده باشد(توجه داشته باشید که هر کدام id های منحصر به فرد خود را دارند). موجودیت غذاها شامل ویژگی های زیر است. دقت داشته باشید که فیلد description به منظور قرار دادن توضیحات اضافی زیر غذا در نظر گرفته شده است.(مثلا : ۱۵۰ گرم سینه مرغ،گوجه کبابی، فلفل کبابی). همچنین دقت داشته باشید که فیلد foodSet مشخص کننده مجموعه غذایی آن رکورد از جدول غذاهاست برای مثال غذای قرمه سبزی جزیی از مجموعه چلو خورشت‌ها است.

```
Food = new schema({
  id: String, // or an auto increment number,
  name:String,
  price:Number, // price of this food in Tomans
  description:String, // optional
  foodSet:String, // set of this food like kabab, khorak, salad
})
```

دسته‌بندی‌ها

مشخص کننده دسته بندی فعالیت رستوران مورد نظر است. هر رستوران میتواند چند نوع دسته بندی داشته باشد و هر دسته بندی میتواند متعلق به چند رستوران باشد. برای مثال رستورانی دسته بندی های غذای ایرانی، غذای دریایی، صبحانه را شامل میشود.

```
Category = new schema({
  id: String, // or an auto increment number,
  name:String,
})
```

آدرس‌ها

هر رستوران یک آدرس دارد که جستجوی کاربر برای پیدا کردن رستوران ها بر اساس آن صورت می پذیرد. هر Schema آدرس مشخصات زیر را داراست.

```
Address = new schema({
  id: String, // or an auto increment number,
  city: String, // e.g. Tehran
  area: String, // e.g. Keshavarz Blvd,
  addressLine:String, // full address text
})
```

فیلدهای area و city برای جست‌وجو در صفحه اصلی که بر مبنای منطقه است استفاده می‌شوند. پس در هنگام ساخت پایگاه‌داده و پر کردن آن با داده‌های تستی حتما توجه کنید که چندین رستوران را در area و city یکسان قرار دهید که بتوان عملکرد کلاینت را مشاهده کرد.

دخیره‌سازی logoرستوران

برای نگه داری لوگو رستوران‌ها پیشنهاد میشود اسم فایل ها با کلید id رستوران یکسان بوده و در یک پوشه نگهداری شود.

مثال: آدرس لوگو مربوط به رستوران با ID: 123456 به صورت http://localhost/posters/123456.jpg می‌باشد.

پیاده‌سازی API

در این بخش به توصیف کلی API های مورد نیاز پروژه می پردازیم. دقت داشته باشید که قسمت دنبال شده پس از عنوان هر API آدرس پیشنهادی و متد پیشنهادی می‌باشد.

دریافت رستوران ها GET - /api/restaurants

کاربرد:

۱- نمایش لیست رستوران‌ها پس از جست‌وجو بر اساس منطقه

GET api/restaurants?area=mirdamad

۲- اعمال فیلتر بر روی رستوران‌های منطقه با استفاده از نوع غذا و به‌روز رسانی لیست رستوران‌ها

GET api/restaurants?area=mirdamad&category=kebab&category=salad

توضیحات:

- کاربر با فراخوانی این رستوران می تواند لیست رستوران های موجود را دریافت کند. دقت داشته باشید که مطمئنا قصد ما نشان دادن کل رستوران های وبسایت به کاربر نیست و میخواهیم با اعمال یک سری قیود لیستی از رستوران ها مقدور به سفارش را برگردانیم.
- با توجه به اینکه در کاربرد دوم ممکن است کاربر با استفاده از چند categoryمختلف رستوران‌ها را فیلتر کند، endpoint ای که طراحی می‌کنید باید توانایی دریافت پارامتر categoryرا به صورت آرایه داشته باشد.

  در فریمورک expressمی‌توانید همانند مثال ذکر شده در بالا عمل کنید.

امتیازی: برای دریافت لیست area ها endpoint ای جداگانه طراحی کنید و در هنگام جست‌وجوی کاربر در صفحه اصلی ابتدا لیست محله را بر اساس عبارت ورودی به کاربر نشان دهید سپس با توجه به منطقه انتخاب شده از لیست توسط کاربر لیست رستوران‌ها را دریافت کنید.

دریافت اطلاعات یک رستوران GET – api/restaurants/:id

کاربرد:

استفاده در صفحه رستوران و نمایش اطلاعات رستوران و منو

GET api/restaurants/shandiz-jordan

توضیحات:

تمام اطلاعات مورد نیاز برای تکمیل صفحه رستوران در بخش کلاینت باید توسط این endpoint دریافت شوند.

امتیازی: فیلد averageRate به صورت خودکار و با توجه به میانگین امتیاز تمامی نظرات محاسبه شود.

دریافت نظرات رستوران GET – api/restaurants/:id/comments

کاربرد:

استفاده در قسمت پایانی صفحه رستوران برای نظرات کاربران در مورد رستوران.

GET api/restaurants/shandiz-jordan/comments

توضیحات:

شامل ارایه ای مرتب شده بر اساس زمان از تمامی مدل های Comment مرتبط با رستوران با کلید :id

ارسال نظر POST – api/restaurants/:id/comments

کاربرد:

ارسال دیدگاه به همراه نام ارسال کننده و امتیاز برای یک رستوران.

توضیحات:

- بدنه درخواست باید شامل اطلاعات لازم برای ایجاد مدل کامنت باشد. بعد از ارسال درخواست یک نمونه از مدل Comment با توجه به فیلدهای داده شده در دیتابیس ذخیره میشود.
- این بخش استفاده‌ای در سمت کلاینت ندارد و کار اضافی در سمت کلاینت نمره اضافه ای نخواهد داشت.

ایجاد رستوران POST – api/restaurants/:id/comments

کاربرد:

ساخت یک رستوران برای نمایش در وب‌سایت

توضیحات:

- بدنه درخواست باید شامل اطلاعات لازم برای ایجاد مدل رستوران باشد. بعد از ارسال درخواست یک نمونه از مدل Restaurant با توجه به فیلدهای داده شده در دیتابیس ذخیره می‌شود.
- توجه کنید که ضروری است هنگام ارسال درخواست، تمامی فیلدهایی که عملکردی در سیستم دارند ارائه شده باشند.
- این بخش استفاده‌ای در سمت کلاینت ندارد و کار اضافی در سمت کلاینت نمره اضافه ای نخواهد داشت.

بخش‌های امتیازی

در صورتی که فرایند کار خود را با استفاده از رویه‌های استاندارد Gitانجام دهید و ثبت کنید، نمره‌ی اضافه به شما تعلق خواهد گرفت.

سایر بخش‌های امتیازی به صورت سبز رنگ در تعریف پروژه آمده اند.

نکات پیاده‌سازی

برای تست عملکرد endpointها اکیدا توصیه می‌شود از REST-Clientهایی مانند Postman یا Insomnia برای ارسال request به برنامه‌تان استفاده کنید.

نکات تحویل پروژه

- پروژه پایانی به صورت انفرادی می‌باشد.
- مهلت ارسال پروژه حداکثر تا ساعت ۲۳:۰۰ روز شنبه ۸ تیر در moodle می‌باشد و تحویل حضوری در روز یکشنبه ۱۰ تیر انجام خواهد شد.
- نام فایل آپلود شده به صورت StudentID.zip(9431000.zip) باشد.

موفق باشید