

به نام خدا



دانشگاه صنعتی امیر کبیر
(پلی تکنیک تهران)

دانشکده مهندسی کامپیوتر
مهندسی نرم افزار ۱

"فاز ۵ پروژه"

موضوع پروژه:
معماری پروژه و طراحی ساختار پایگاه داده

استاد درس: دکتر کلباسی

مهلت تحویل: جمعه ۵ دی ۱۴۰۴

نیم سال اول ۱۴۰۴

مقدمه

پروژه توسعه وبسایت ایران‌شناسی و ایران‌گردی یک پلتفرم جامع برای معرفی جاذبه‌های فرهنگی، تاریخی و طبیعی ایران است که بر پایه معماری میکروسرویس‌ها پیاده‌سازی می‌شود. این معماری، که از اصول توزیع‌شده و ماژولار پیروی می‌کند، اجازه می‌دهد تا سیستم به صورت مقیاس‌پذیر، انعطاف‌پذیر و قابل نگهداری توسعه یابد. در این رویکرد، هر میکروسرویس به عنوان یک واحد مستقل عمل می‌کند و مسئولیت خاصی را بر عهده دارد. تیم‌های توسعه می‌توانند به طور موازی بر روی سرویس‌های خود کار کنند، که این امر سرعت توسعه را افزایش و ریسک شکست کلی سیستم را کاهش می‌دهد. فاز فعلی پروژه، بر دو بخش اصلی تمرکز دارد: **طراحی پایگاه داده و مشخص کردن معماری میکروسرویس**.

این فاز حیاتی است زیرا پایه و اساس فنی پروژه را شکل می‌دهد و تضمین می‌کند که سیستم از ابتدا با اصول مدرن مهندسی نرم‌افزار همخوانی داشته باشد. در بخش طراحی پایگاه داده، تیم‌ها مسئول تعریف ساختار داده‌های اختصاصی میکروسرویس خود هستند، به طوریکه استقلال سرویس‌ها حفظ شود و در عین حال امکان تعامل ایمن و کارآمد بین آن‌ها فراهم گردد. این بخش شامل اصول کلی، استانداردهای طراحی، **ER Diagram** و توضیحات جدول‌ها می‌شود تا یکپارچگی داده‌ها تضمین شود و مشکلات آینده مانند مهاجرت داده‌ها یا مقیاس‌پذیری داده‌ها به حداقل برسد.

یکی از بخش‌های اصلی در طراحی هر نرم‌افزار، معماری نرم‌افزار است که چارچوب کلی سیستم را مشخص می‌کند. معماری نرم‌افزار کمک می‌کند اجزای مختلف سیستم به درستی کنار هم قرار بگیرند و بتوانند با هم به خوبی کار کنند. انتخاب معماری مناسب باعث می‌شود نرم‌افزار عملکرد بهتری داشته باشد، راحت‌تر توسعه پیدا کند و نگهداری آن در آینده ساده‌تر باشد. به همین دلیل، آشنایی با معماری نرم‌افزار و انواع آن برای توسعه‌دهندگان اهمیت زیادی دارد.

طراحی پایگاه داده

پروژه توسعه وبسایت ایران شناسی و ایران گردی بر پایه معماری میکروسرویس ها طراحی شده است. در فاز فعلی، هر تیم مسئول طراحی پایگاه داده اختصاصی میکروسرویس خود می باشد. این راهنما به دو بخش اصلی تقسیم می شود:

۱. چارچوب و قوانین کلی طراحی پایگاه داده: اصول و استانداردهایی که تمامی تیم ها ملزم به رعایت آن ها هستند تا یکپارچگی، کارایی و امنیت سیستم حفظ شود.

۲. ER Diagram و توضیحات جدول ها: راهنمایی برای طراحی و ترسیم نمودار روابط موجودیت ها (Entity-Relationship Diagram) به منظور مدل سازی ساختار دیتابیس، همراه با توضیحات اضافی در مورد جدول ها. با توجه به احتمال نیاز تیم ها به تعامل با دیتابیس های یکدیگر در آینده، تأکید بر اصل "Database per Service" وجود دارد؛ به این معنی که تعاملات باید از طریق API ها، Message Broker ها یا الگوهای Event-Driven انجام شود تا وابستگی مستقیم ایجاد نشود.

بخش اول: چارچوب و قوانین کلی طراحی پایگاه داده

۱. اصول کلی معماری دیتابیس در میکروسرویس‌ها

– استقلال پایگاه داده (Database per Service): هر میکروسرویس باید دیتابیس مستقل خود را داشته باشد تا از مشکلات ناشی از پایگاه داده اشتراکی جلوگیری شود.

– انتخاب نوع پایگاه داده: بر اساس نیاز سرویس، از دیتابیس‌های رابطه‌ای (مانند PostgreSQL یا MySQL) برای داده‌های ساخت‌یافته یا NoSQL (مانند MongoDB) برای داده‌های نیمه‌ساخت‌یافته استفاده شود.

– توجه به تعاملات آینده: برای دسترسی به داده‌های سرویس‌های دیگر، از APIهای RESTful یا GraphQL بهره ببرید. از الگوهایی مانند CQRS یا Event Sourcing برای همگام‌سازی داده‌ها استفاده کنید.

۲. استانداردهای نام‌گذاری و ساختار

– نام‌گذاری جدول‌ها و ستون‌ها: جدول‌ها با پیشوند نام سرویس آغاز شوند (مثال: users_profiles). از snake_case برای نام‌ها استفاده شود (مثال: user_id).

– انواع داده‌ها: از UUID یا BigInt برای IDها، TIMESTAMP WITH TIME ZONE برای تاریخ‌ها و TEXT برای محتوای طولانی استفاده شود.

– کلیدها و محدودیت‌ها: هر جدول دارای Primary Key باشد. Foreign Keyها برای حفظ یکپارچگی داده‌ها در صورت لزوم اعمال شوند.

۳. بهینه‌سازی عملکرد

– Indexing: ایندکس بر روی ستون‌های پرستفاده ایجاد شود. می‌توانید از Composite Indexes برای کوئری‌های پیچیده استفاده کنید.

– **Sharding و Partitioning**: برای داده‌های حجیم، در صورت نیاز، جداول پایگاه داده را بر اساس معیارهایی مانند تاریخ یا مکان جغرافیایی پارتیشن‌بندی کنید.

۴. امنیت و پشتیبان‌گیری

– **امنیت داده‌ها**: داده‌های حساس رمزنگاری شوند. برای مثال، گذرواژه‌ها (passwords) به صورت hash ذخیره شوند.

– **Recovery و Backup**: برنامه‌ریزی برای پشتیبان‌گیری منظم و بازیابی در شرایط بحرانی انجام شود.

۵. نسخه‌بندی و مستندسازی

– **مستندسازی**: schema در فرمت SQL یا YAML مستندسازی شده و در ریپازیتوری مشترک قرار گیرد.

۶. نکات خاص برای این پروژه

– **داده‌های جغرافیایی**: از PostGIS برای ذخیره مختصات جغرافیایی استفاده شود.

– **چندزبانه بودن**: فیلدهای متنی برای زبان‌های فارسی و انگلیسی طراحی شوند.

– **مقیاس‌پذیری**: کوئری‌ها برای حجم بالای کاربران بهینه‌سازی شوند.

بخش دوم: ER Diagram و توضیحات جدول‌ها

۱. اجزای اصلی ER Diagram

- Entities: موجودیت‌های اصلی (جداول)
- Attributes: ویژگی‌های درون هر موجودیت
- Relationships: روابط میان موجودیت‌ها

۲. مراحل طراحی ER Diagram

- ۱) شناسایی Entities: بر اساس نیاز میکروسرویس، موجودیت‌ها را لیست کنید.
- ۲) تعریف Attributes: ویژگی‌های ضروری هر موجودیت را مشخص کنید.
- ۳) تعریف Relationships: روابط را بر اساس سناریوها مدل کنید.
- ۴) ER Normalization: را نرمال‌سازی کنید.
- ۵) رسم نمودار: از ابزارهایی مانند Enterprise Architect، Draw.io یا Lucidchart استفاده شود.
- ۶) مستندسازی: توضیحات هر بخش اضافه شود.

۳. توضیحات جدول‌ها

علاوه بر طراحی ER Diagram، هر تیم ملزم است توضیحات کاملی در مورد هر جدول پایگاه داده خود ارائه دهد. این توضیحات باید به صورت ساخت‌یافته تهیه شود تا درک ساختار پایگاه داده برای تیم‌های دیگر آسان‌تر گردد. هدف از این بخش، افزایش شفافیت، تسهیل تعاملات آینده (مانند استفاده از API‌ها برای دسترسی به داده‌ها) و جلوگیری از ابهامات در نگهداری سیستم است.

توضیحات هر جدول باید شامل بخش‌های زیر باشد:

- **اسم جدول:** نام کامل جدول بر اساس استانداردهای نام‌گذاری.

- **توضیح کلی:** توصیف مختصر از نقش جدول در میکروسرویس، داده‌هایی که ذخیره می‌کند و ارتباط آن با سایر بخش‌های سیستم.

- **قیدها، محدودیت‌ها و توضیحات هر ویژگی:** برای هر ستون (ویژگی) جدول، نام ستون به همراه توضیح مختصر آورده شود. اگر این ستون به سرویس دیگری لینک می‌شود، توضیح دهید چگونه تعامل انجام می‌شود (مثال: "Foreign Key" به جدول locations در سرویس مکان‌ها با دسترسی از طریق API).

- **داده‌های نمونه:** حداقل ۲-۳ ردیف داده نمونه (Sample Data) برای هر جدول ارائه دهید تا ساختار واقعی داده‌ها نشان داده شود.

نکته: توضیحات جدول‌ها باید با ER Diagram همخوانی داشته باشند. مثلاً اگر در ER یک رابطه یک به چند وجود دارد، در توضیحات قیدها به آن اشاره شود.

این بخش کمک می‌کند تا تیم‌ها بتوانند بدون نیاز به کد، ساختار پایگاه داده یکدیگر را درک کنند و در فازهای بعدی (مانند ادغام سرویس‌ها) مشکلات کمتری پیش آید.

نمونه توضیحات جدول پایگاه داده

۱. نام جدول:

City

۲. توضیح کلی:

این جدول اطلاعات یک شهر، از جمله نام و شناسه کشور را نگه می‌دارد. هر شهر دارای یک شناسه یکتاست.

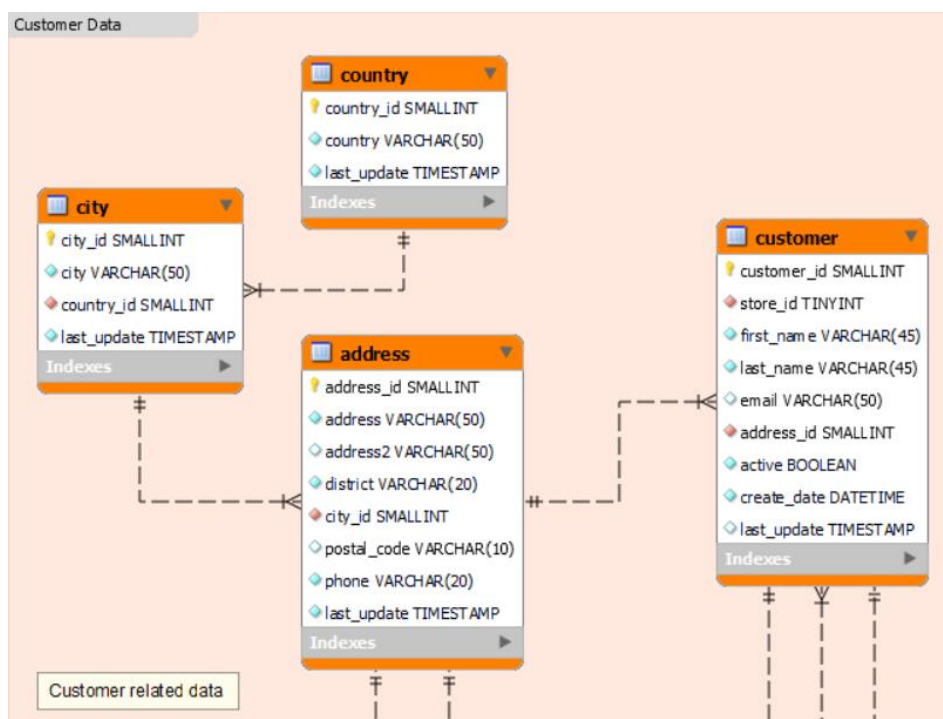
۳. قیدها و توضیحات هر ویژگی:

ویژگی	توضیح
city_id	شاخص یکتای هر شهر که توسط دیتابیس تولید می‌شود.
city	نام شهر. نمی‌تواند خالی و تکراری باشد.
country_id	شناسه کشوری که شهر در آن قرار دارد. نمی‌تواند خالی باشد.
last_updated	آخرین زمانی که مقادیر سطر تغییر کردند.

۴. داده نمونه (Sample Data):

city_id	city	country_id	last_updated
1	Tehran	5	2000-05-20
2	Rasht	12	1999-11-10

نمونه ER Diagram



توجه کنید که روابط، قیدها و محدودیت‌ها (به کمک علائم کنار هر ویژگی موجودیت) و نوع داده هر ویژگی در این نمودار مشخص شده‌اند. اگر هر کدام از این بخش‌ها قابل نمایش در ER Diagram رسم شده شما نیست، باید در بخش توضیحات جداول آورده شوند.

نکته: نمودار ER (Entity-Relationship Diagram) در این فاز صرفاً با هدف مدل‌سازی ساختار پایگاه داده ترسیم می‌شود و تمرکز آن بر موجودیت‌ها (جداول)، ویژگی‌ها (ستون‌ها)، روابط بین موجودیت‌ها، کلیدهای اصلی و خارجی، قیود و نوع داده‌ها است. این نمودار نمایانگر نحوه ذخیره‌سازی و سازمان‌دهی داده‌ها در پایگاه داده هر میکروسرویس بوده و مستقیماً به طراحی Schema دیتابیس مربوط می‌شود. در مقابل، Class Diagram که در فاز بعدی تهیه خواهد شد، برای مدل‌سازی ساختار نرم‌افزار و طراحی شیء‌گرا (کلاس‌ها، فیلدها/متدها، روابط ارث‌بری، وابستگی‌ها و الگوهای طراحی) به کار می‌رود. بنابراین، موجودیت‌های ER الزاماً معادل کلاس‌های سیستم نیستند و نباید روابط دیتابیسی (مثل Foreign Key) با روابط شیء‌گرایانه (مثل inheritance/association/composition) اشتباه گرفته شوند.

در این راستا، [این لینک](#) می‌تواند به عنوان یک منبع تکمیلی مورد استفاده قرار گیرد.

معماری پروژه

معماری نرم‌افزار به ساختار کلی سیستم اطلاق می‌شود که شامل نحوه سازماندهی اجزای مختلف نرم‌افزار، ارتباطات و تعاملات بین این اجزاء، و شیوه‌ای است که سیستم به طور کلی عمل می‌کند. معماری نرم‌افزار یکی از مهم‌ترین بخش‌های هر پروژه توسعه نرم‌افزار است زیرا تصمیمات معماری به طور مستقیم بر عملکرد، مقیاس‌پذیری، قابلیت نگهداری، امنیت و توسعه‌پذیری سیستم تأثیر می‌گذارند. انتخاب معماری مناسب می‌تواند به موفقیت پروژه کمک کند و عدم توجه به آن می‌تواند منجر به مشکلات جدی در مراحل بعدی توسعه شود.

انواع مختلفی از معماری‌های نرم‌افزاری وجود دارند که متناسب با نیازهای مختلف پروژه انتخاب می‌شوند. به طور مثال:

- **معماری یکپارچه (Monolithic Architecture):** در این نوع معماری، تمامی اجزای سیستم در یک واحد واحد پیاده‌سازی می‌شوند و به صورت یکپارچه عمل می‌کنند. این معماری ساده و مناسب پروژه‌های کوچک است.
- **معماری میکروسرویس (Microservices Architecture):** در این معماری، سیستم به مجموعه‌ای از سرویس‌های کوچک و مستقل تقسیم می‌شود که هر کدام وظیفه خاصی دارند و از طریق API ها با یکدیگر ارتباط برقرار می‌کنند. این معماری برای پروژه‌های بزرگ و مقیاس‌پذیر مناسب است.
- **معماری لایه‌ای (Layered Architecture):** در این مدل، سیستم به چندین لایه تقسیم می‌شود که هر لایه مسئول یک بخش خاص از عملیات سیستم است، مانند لایه ارائه، لایه منطق تجاری، و لایه داده.

انتخاب معماری متناسب با نیازمندی‌ها

در این فاز از پروژه، از شما انتظار می‌رود که معماری متناسب با نیازمندی‌های پروژه خود را انتخاب کنید. توجه کنید که معماری کلی استفاده شده در پروژه میکروسرویس است ولی لازم است شما برای پیاده‌سازی سرویس انتخابی خود نیز یک معماری انتخاب کنید.

هنگام انتخاب معماری، باید به نکات زیر توجه داشته باشید:

۱. نیازمندی‌های مقیاس‌پذیری: آیا سیستم باید توانایی پردازش حجم بالای درخواست‌ها را داشته باشد؟
۲. سطح پیچیدگی پروژه: آیا سیستم نیاز به تقسیم‌بندی پیچیده به بخش‌های مستقل دارد؟
۳. نیاز به انعطاف‌پذیری: آیا ممکن است نیاز به تغییرات سریع و ساده در سیستم باشد؟
۴. امنیت: آیا سیستم به امنیت بالایی نیاز دارد؟
۵. توانایی نگهداری و توسعه: آیا نیاز به به‌روزرسانی یا اضافه کردن قابلیت‌های جدید به راحتی وجود دارد؟

مواردی که نیاز است تحویل دهید

فایل pdf حاوی موارد زیر:

- توضیحات پایگاه داده و ER Diagram
- معماری انتخاب شده و دلایل انتخاب آن و توضیحات مربوطه

نکات تحویل

- دقت کنید که تمام مراحل این فاز پروژه به صورت گروهی دنبال شود و تمام اعضا در تحقیق و انتخاب موضوع نقش داشته باشند.
- فایل PDF خواسته شده را سرگروه در سایت courses آپلود کند.
- فرمت نام فایل PDF به صورت P5_GroupName.pdf باشد که در آن GroupName نام گروه است.
- ارائه‌ی فایل مرتب و منظم و رعایت قالب‌های درست در نوشتن فایل نهایی شما، بخشی از نمره‌ی شما را تشکیل می‌دهد.
- برای تحویل دادن هر فاز از پروژه تنها آپلود یکی از اعضای گروه کافی‌ست.
- حتما شماره دانشجویی تمام اعضای گروه را در انتهای فایل pdf ذکر کنید.