Technical Documentation

معماری پروژه

در معماری این پروژه از رویکرد MVC استفاده شده. به این صورت که این پروژه شامل سه بخش اصلی Models، Views و Controllers است.

در این پروژه سعی شده تا تمامی بخش ها به ماژولار ترین حالت ممکن پیاده سازی شوند و قسمت رابط کاربری از بخش back end و سایر بخش های از یکدیگر جدا باشند و کم ترین وابستگی را به هم داشته باشند.

بخش Models

این بخش یک سیستم ORM است و تمامی ارتباط با database توسط این بخش اجرا می شود. در این بخش سعی شده سیستمی مانند Eloquent در فریم ورک Laravel پیاده سازی شود.

در این سیستم به ازای هر table در database یک Model وجود دارد که در آن کلاسی متناظر با آن table قرار دارد و وظیفه تعامل با آن table بر عهده این کلاس است.

کلاس های model همه از یک interface به نام Model ارث بری می کنند. و هر object از این کلاس ها متناظر با یک سطر از یک tabel در database است و property های هر کلاس ستون های table هستند.

هر کلاس مدل مطابق Model interface شامل static method هایی برای اجرای عملیات های Create، Read، Update، Delete و عملیات های مشابه دیگری است. همچنین هر کلاس instance method هایی دارد که برای اجرای عملیات هایی مخصوص جدولی که با آن در ارتباط هستند تعریف شده اند.

در این پروژه مدل های زیر وجود دارد:

مدل User برای ارتباط با جدول user و انجام عملیات های مرتبط با کاربر

مدل Menu برای ارتباط با جدول menus و انجام عملیات های مربوط به منو ها

مدل Order برای ارتباط با جدول orders و انجام عملیات های مربوط به سفارش های کاربران

مدل Food.py مدلی است که با table، foods در Database تبادل اطلاعات می کند. این فایل حاوی کلاسی به نام Food است که با استفاده از استاتیک متود های خود با دیتابیس ارتباط می گیرد.

مدل GiftCard.py با table، gift_cards در Database ارتباط می گیرد. این فایل نیز با استاتیک متد های خود اطلاعات را با دیتابیس رد بدل می کنند.

فایل Restaurant.py در واقع یک مدل نیست و به همین دلیل از کلاس Model ارث بری نکرده است. این فایل اطلاعات ثابت رستوران را Get و Set می کند. و در فایلی به نام Restaurant.json در پوشه Constants ذخیره می کند.

مدل Comment.py کامنت ها را ذخیره می کند.

مدل Message.py پیام های چت مدیران را در دیتابیس ذخیره می کند.

بخش View ها

ویو ها فایل هایی هستند که بخش رابط کاربری را تشکیل می دهند. هر صفحه از پروژه یک view است که با اجرای آن view به نمایش در می آید. پوشه view ها شامل بخش user است که وظیفه اجرای رابط کاربری سمت کاربر و بخش admin که وظیفه اجرای رابط کاربری بخش مدیر و همچنین view های عمومی است.

فایل های ویو پروژه با استفاده از qtdesingner طراحی شده اند. فایل ui. هر صفحه در کنار فایل py. آن قرار دارد که باعث می شود برای اضافه کردن component های جدید به صفحه، کار راحت تری در پیش داشته باشیم.

هر فایل view شامل دو بخش است بخش event ها و بخش ui.

بخش ui شامل کلاس Ui_MainWindow است که توسط نرم افزار qt designer ساخته میشود و دارای متد setupUi است که یک window که یک آبجکت از نوع QMainWindow به عنوان آرگومان دریافت می کند و ui طراحی شده را روی آن اجرا می کند.

بخش event ها شامل متد هایی است که وظیفه پیاده سازی تعامل با صفحه را دارند. این بخش برای مینیمم کردن ارتباط منطق با ui وجود دارد و شامل چند متد است.

متد init که پس از load شدن صفحه اجرا می شود.

متد هایی که هر کدام به یکی از button های صفحه متصل هستند و در صورت کلیک کردن روی آن دکمه اجرا می شوند. این متد ها از توابع کنترلر ها استفاده کرده و عملیات های موجود در هر صفحه را انجام می دهند.

متد های getter و setter که بخش های داینامیک صفحه را ست می کنند یا اطلاعات فرم ها را از QlineEdit ها می گیرند. و در متد های متصل به button ها استفاده می شوند.

در هر صفحه مربوط به admin ابتدا چک می شود که کاربری که در صفحه حضور دارد login کرده باشد و دارای دسترسی admin باشد. در ادامه کامپوننت های صفحه به کمک فایل های کنترلر مروطه مقدار دهی می شوند. و در نهایت هم عملکرد هر دکمه و component صفحه به آن داده می شود.

بخش Controller

کنترلر ها وظیفه اجرای منطق پروژه را دارند. رابط بین Model ها و View ها هستند. کنترلر ها Model ها را import کرده و در تعدادی static method عملیات های مورد نظر را پیاده سازی کرده و در view ها import می شوند.

در این پروژه کنترلر های زیر وجود دارند

CartController

که وظیفه اجرای عملیات های مربوط به سبد خرید را دارد.

MenuController

که وظیفه اجرای عملیات های مورد نیاز صفحه منو ها را دارد

OrderController

که وظیفه ساخت سفارش ها را دارد

UserInfoController

که وظیفه update کردن اطلاعات کاربران را دارد.

GiftCardController

در این فایل عملیات پیشنهاد یک گیفت کارت به کاربر های خوب و همچنین عملیات ارسال گیفت کارت ها و همچنین راستی آزمایی code وارد شده از طرف کاربر انجام می شود.

سیستم Routing

سیستم routing و جابه‌جایی بین صفحه ها توسط دو فایل Window و routes انجام میشود. هر view برای اتصال به پروژه باید در فایل routes ثبت شود و پس از ثبت بدون نیاز به هیچ تغییری در فایل view با استفاده از تابع Redirect در کلاس Routing در فایل window قابل دسترسی است.

کلاس Window در فایل Window نیز وظیفه ساخت هر window را دارد. و ابتدا فایل view صفحه فعلی را لود کرده و سپس یک نمونه از کلاس ui_MainWindow ایجاد کرده و با استفاده از متد setupUI رابط کار بری را روی window اعمال می کند.

کلاس Transfer هم شامل متد هایی برای انتقال دیتا بین دو صفحه است.

سیستم Authentication

سیستم احراز هویت به صورت role based پیاده سازی شده و در کنترلر Auth هندل می شود. عملیات های login و signup و log out در این کنترلر پیاده سازی شده اند. این کنترلر همواره اطلاعات کابری که login کرده است را ذخیره می کند و با متد GetUser می توان آبجکت مدل User برای کاربر وارد شده را دریافت کرد.

سیستم Validation

اعتبار سنجی همه input ها در کنترلر validationController انجام می شود.

این کنترلر شامل کلاس های validator است و در هر کلاس متد هایی برای اعتبار سنجی input های مرتبط وجود دارد.

بخش DataBase

فولدر DataBase شامل فایل sqlite است که دارای متد هایی برای اجرای عملیات های CRUD رو دیتابیس است. این فایل تنها در Model ها برای ارتباط با دیتابیس استفاده میشود.

در این فولدر فایل TablesData هم وجود دارد که شامل دستور های sql برای ایجاد هر table است.

کتابخانه ها (Lib)

در این بخش ماژول ها و ابزار هایی برای انجام عملیات های عمومی وجود دارد.

:DateTools

شامل متد هایی برای کار با تاریخ و عملیات های شخصی سازی شده مربوط به ماژول datatime

:Email

پیاده سازی سیستم ارسال ایمیل

:Messages

سیستم notification و ارسال پیغام های خطا و ... در هنگام اجرای برنامه به کاربر.

:Image.py

در این فایل دو عملیات ذخیره عکس از یک مسیر و از یک آدرس اینترنتی انجام می شود. متود GetFromDirectory(path:str) در واقع یک مسیر را می گیرد و تصویر در آن مسیر را در پوشه Resources/Images کپی می کند و آدرس Resources/Images/name.jpg را بر می گرداند. اگر مسیر داده شده وجود نداشته باشد یا ارور دیگری رخ دهد، این تابع مقدار None بر می گرداند. اگر تصویری که مسیر آن داده شده، دقیقا با یکی از عکس های Resources/Image یکی باشد، عکس را کپی نخواهد کرد و صرفا مسیر را بر می گرداند.

Questions.py

در این فایل دو عملیات صورت می گیرد. یکی پرسش به صورت ok/cancel و یکی به صورت yes/no. برای این کار از کتابخانه tkinter استفاده شده و متود های آن مقدار bool نتیجه را بر می گردانند.

SpellCorrecting.py

در این فایل با استفاده از کتابخانه textblob سیستمی برای تصحیح غلط املایی پیاده سازی شده است. متود اصلی این کلاس KeyPressHandler است که با connect کردن آن به یک lineEdit یک صفحه pyqt5، غلط های املایی آن را تصحیح خواهد کرد.

FoodSearch.py

این فایل با استفاده از کتابخانه difflib میزان شباهت بین دو رشته را اندازه گیری می کند و میزان شباهت را به صورت عددی بین 0 تا 1 بر می گرداند. متود اصلی این کلاس find است که با گرفتن نوع سرچ، با توجه به آن attr در دیتابیس سرچ کرده و نتیجه را به صورت لیستی از Food ها بر می گرداند.

سیستم Initialize

فایل initialize که وظیفه اجرای عملیات هایی در هنگام آغاز اجرای app را دارد که شامل ایجاد و مطمئن شدن از وجود جدول ها در دیتا بیس و افزودن اطلاعات Default به سیستم است

صفحه Chat

در این صفحه ابتدا یک textEdit قرار داده شده و آن را ReadOnly کرده ایم. پیام های چت در این widget قرار می گیرد. با وارد شدن به این صفحه تمام پیام ها از دیتابیس گرفته می شود و نمایش داده می شود. در این صفحه یک checkbox به نام Assistant وجود دارد که با استفاده از کتابخانه SpellCorrecting.py در پوشه Lib غلط املایی admin را اصلاح می کند. این checkbox با فعال شدن lineEdit اصلی تایپ کاربر را به متود KeyPressHandler از فایل SpellCorrecting.py وصل می کند و با غیر فعال شدن lineEdit را از آن جدا می کند. در نهایت با فشردن دکمه Enter یا کلیک کردن بر روی Send پیام ارسال می شود و در table، messages دیتابیس ذخیره می شود.

بات تلگرام

این رباط با استفاده از کتابخانه رایگان python-telegram-bot نوشته شده است و با استفاده از پروتکل API پیاده سازی شده است.

سیستم رباط به این شکل است: فایل اصلی اجرای رباط Bot.py است که در آن کلاس Telegram وجود دارد. این کلاس وظیفه مقدار دهی کردن اولیه رباط با دارد. علاوه بر این فایل، در هر پوشه فایل هایی وجود دارد که عملیات مربوط به خود را انجام می دهد. برای مثال پوشه Chat حاوی متود های هندل کننده command های مربوط به چت است. و همچنین Login عملیات Authentication کاربر را به عهده دارد.

این رباط کاملا به صورت ماژولار پیاده سازی شده و همین موضوع قابلیت توسعه فوق العاده ای به آن می دهد. با اضافه کردن یک پوشه جدید و اضافه کردن CommandHandler و MessageHandler ها مورد نیاز می توان رباط را توسعه داد. تنها محدودیت زمانی است که در اختیار داریم.