به نام خدا

پروژه ی مهندسی نرم‌افزار - فاز سوم

اعضای گروه:

مهزیار مومنی 9412358036

مهرداد سلحشوری 9612358022

کیارش نوروزی 9612358042

محمدعلی افتخاری 9612358003

سید فرهاد حسینی 9612358016

محمدرضا حاجی بابا 9612358014

مقدمه :

معماری نرم‌افزار از کلیدی‌ترین بخش‌های تولید نرم‌افزار است.معماری نرم‌افزار در واقع انتخاب یک ساختار کلی برای پیاده‌سازی یک پروژه نرم‌افزاری بر مبنای مجموعه‌ای از نیازهای کاربری و تجاری یک سیستم نرم‌افزاری است تا هم بتوان کاربردهای مورد نظر را پیاده‌سازی کرد و هم بتوان کیفیت نرم‌افزار، تولید آن و نگهداری آن را نیز بهینه کرد و سرعت بخشید.

یافتن نیازهای غیرعملکردی:

امنیت: در سیستم ما امنیت دارای دو بخش است:

۱. امنیت اطلاعات مشتری‌ها: به دلیل آنکه اطلاعات پایه‌ای و حساس مردم در این سامانه ثبت می‌شود و دسترسی کلان به این اطلاعات می‌تواند مشکلات زیادی را در کل سیستم‌های اداری کشور بوجود آورد، امنیت اطلاعات کاربران و مشتریان بسیار حائز اهمیت است. همچنین اگر مشکل ایجاد شده باعث از بین رفتن داده ها بشود پیمانکار موظف به ریکاوری داده ها می باشد.

۲. تعیین سطح دسترسی کارفرما، مدیران و مسئولین سیستم: در راستای برقراری امنیت فوق، و جلوگیری از باقی مشکلات سیستمی و امنیتی، بهتر است کاربران سیستم را که به کارفرما، مدیران و مسئولین موجود در مراکز پیشخوان خدمت تقسیم بندی می‌شوند، هر کدام در سطوح دسترسی مختلفی به امکانات و اطلاعات ذخیره شده دسترسی پیدا کنند.

کارایی: در RFP بیان شده آپدیت لحظه‌ای آمارها و گزارش‌ها، افزایش سرعت رسیدگی به درخواستها و ذخیره سازی اطلاعات به صورت بهینه (عدم نیاز به فضای فیزیکی) در ساخت برنامه اهمیت دارد پس برنامه ما باید کارایی بالایی داشته باشد.

قابل اعتماد: کارهای حیاتی و مهم بین دولت و مردم به صورتی است که محدودیت زمانی در آن وجود دارد، سیستم حتی الامکان باید سریع و قابل اعتماد باشد و در هر لحظه بتواند پاسخگوی نیاز ها باشد و مشکلات احتمالی سیستم سریع رفع گردد .(این موضوع مهم در RFP ذکر نشده بود)

قابلیت نگهداری: با توجه به این که در RFP گفته شده وظیفه پشتیبانی از سیستم تا 6 ماه به طور کامل با سازندگان است و همچنین نرم افزار باید ماهیانه تست شده و ایرادات آن برطرف شود قابلیت نگهداری در سیستم ما اهمیت زیادی دارد. همچنین اگر مشکل ایجاد شده باعث از بین رفتن داده ها بشود پیمانکار موظف به ریکاوری داده ها می باشد.

منعطف بودن: سیستم پیشخوان به دلیل این که باید با استفاده های جدید خود را تطبیق دهد باید منعطف باشد. زیرا که این سیستم رابط بین دولت و مردم است و هر خدمت جدیدی که نیاز به این تعامل دارد باید در این سیستم اضافه شود.

قابل مدیریت: به دلیل آنچه که در RFP گفته شده که پیمانکار موظف می‌باشد گزارشات پیشرفت پروژه را هر هفته به مدیر فناوری اطلاعات شرکت ارائه بدهد، بر آنیم که این نیازمندی غیر عملکردی را جزئی از خواص سیستم در نظر بگیریم.

قابل استفاده: راحتی مشاهده و ویرایش اطلاعات مشتریان جز مواردی بود که در RFP ذکر شده بود و انتظار می‌رود که معماری ارائه شده دارای این ویژگی باشد.

قابل استفاده مجدد: چون وجه اشتراک زیادی بین بخش‌های مختلف خدمات پیشخوان وجود دارد، قابل استفاده مجدد بودن منابع می‌تواند سرعت توسعه و پیشرفت پروژه را به ارمغان آورد.

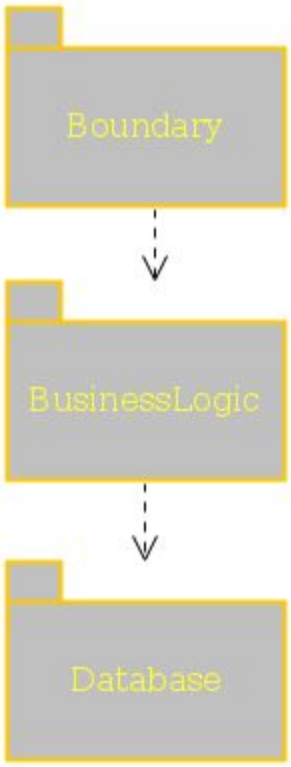
اقتصادی: با توجه به این که صفاتی نظیر کارایی و امنیت برای ما اهمیت زیادی دارند به دلیل trade-off آنها با هزینه ، هزینه جز اولویت های ما نیست.

معماری سیستم:

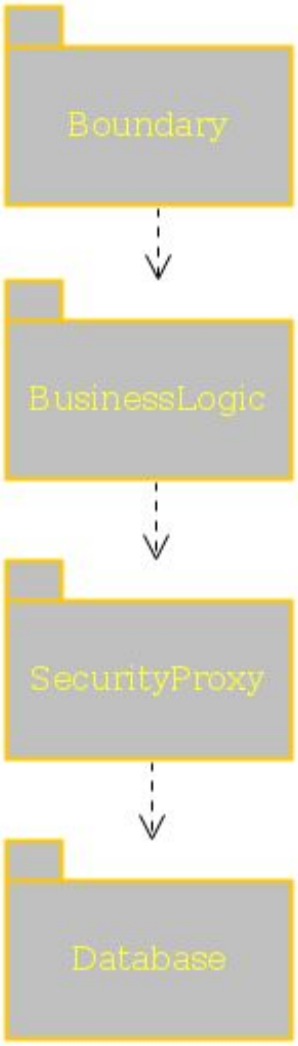
در ابتدا سبک معماری سیستم را انتخاب می‌کنیم که طی بررسی‌ها انجام شده سبک معماری Subsystems با نیازمندی‌های غیرعملکردی سیستم ما مناسب‌تر است زیرا که در این سبک، قابلیت reuse کردن کامپونتت‌ها بیشتر است و از طرفی برنامه‌ریزی برای انجام پروژه راحتتر است، پیچیدگی طراحی کمتری دارد و Maintainability بیشتری را داراست.

حال برای جدا سازی Subsystem های موجود در سیستم از Layering and Partitioning استفاده می‌کنیم.

۱. از معماری سه لایه‌ای شروع می‌کنیم، همانطور که مشاهده می‌شود این معماری نمی‌تواند پاسخگوی نیازهای غیرعملکردی سیستم ما باشد و شروع به گسترش دادن معماری می‌کنیم.

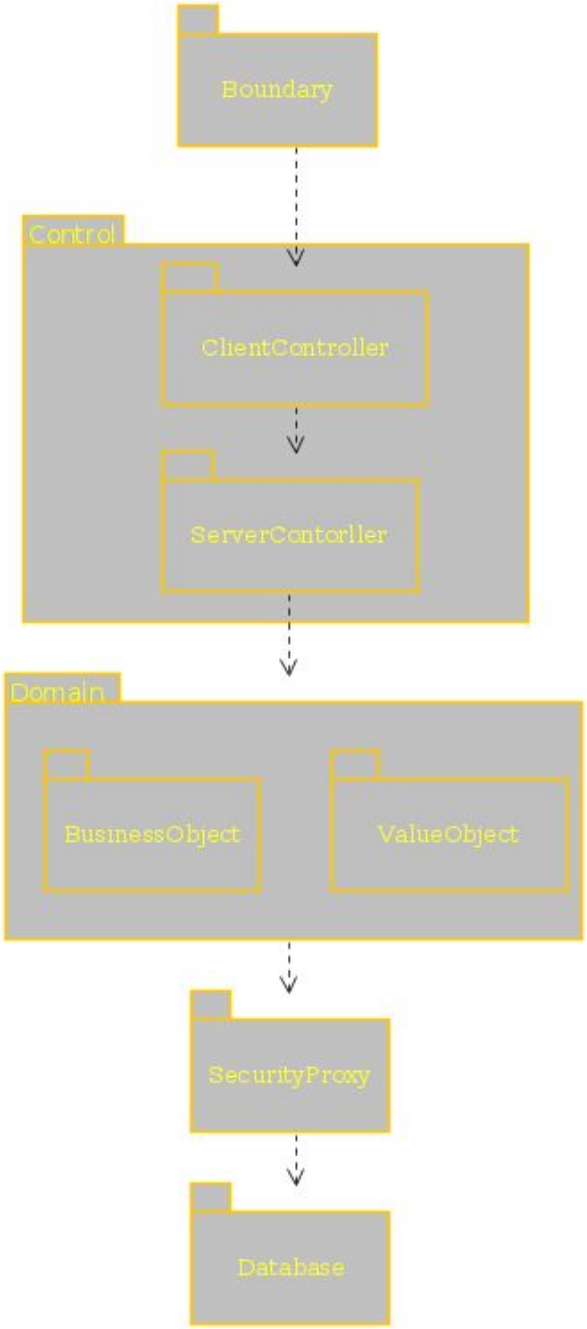


۲. حال برای حل مشکل امنیت سیستم که در بخش فوق به دو قسمت امنیت اطلاعات مشتری‌ها و برقراری سطح دسترسی تقسیم شده است، زیرسیستم SecurityProxy ای بعد از کنترلر قرار می‌دهیم. از طرفی هم ارتباط بین لایه‌ها را Close در نظر می‌گیریم تا امنیت بهتری را در سطح سیستم داشته باشیم.



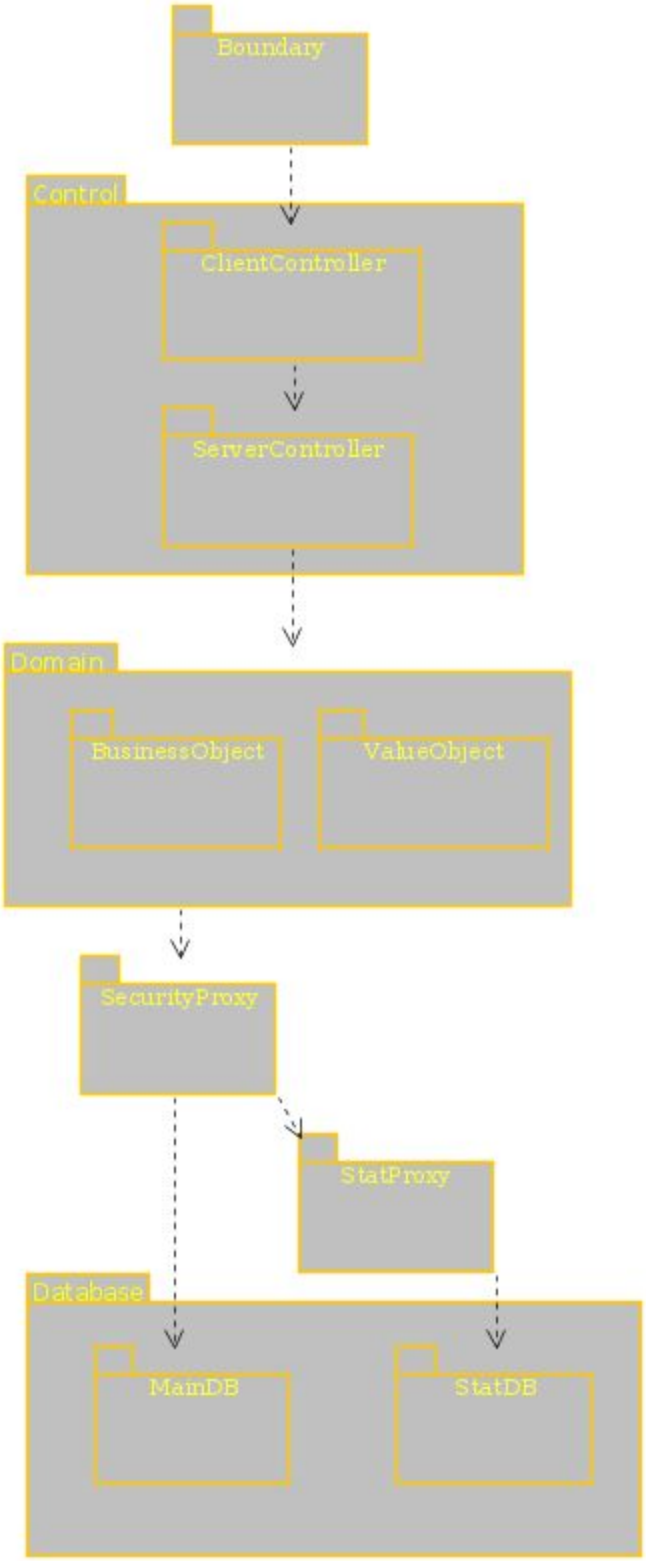
۳. برای افزایش کارایی سیستم از نظر سرعت، لایه‌ی BusinessLogic تبدیل به دو لایه‌ی Controller و Domain می‌شود که از Value Object ها در Domain استفاده می‌شود تا هر بار اطلاعات غیر مفید اضافی بین لایه رد و بدل نشود و صرفا اطلاعاتی که نیاز داریم را بین لایه ها منتقل کنیم و این امر سبب بهبود سرعت و کارایی می‌شود و همچنین Controller را به دو بخش ServerController و

ClientController تقسیم بندی می‌کنیم تا بخشی از کنترل سمت کاربر و بخشی از کنترل سمت سرور باشد. بخشی که سمت سرور است وظیفه امنیت را برعهده دارد و چون اشیای ما به لحاظ امنیتی مهم هستند لزوما باید از کلاس کنترلی سمت سرور به آنها دسترسی داشته باشیم.



ضمنا از پروکسی‌ای جهت Cache کردن اطلاعات مرتبط به موارد آماری استفاده می‌شود(StatProxy) تا نیازی به دسترسی مکرر به دیتابِیس MainDB جهت بدست آوردن اطلاعات آماری نداشته باشیم از طرفی

نیز StatDB ای قرار داده شده است که این اطلاعات آماری طبق یک Interval زمانی‌ای در دیتابیسی ذخیره شود.



۴. حال برای برقراری امنیت، قابلیت نگهداری و حفظ قابلیت اطمینان سیستم، بک‌آپی از دیتابیس را در معماری سیستم قرار می‌دهیم و دسترسی بکاپ و خود دیتای اصلی فقط از سمت SecurityProxy امکان پذیر است.