

به نام خدا

پروژه درس معماری نرم افزار

نام و نام خانوادگی: امیر حسین یزدان پرست

استاد راهنما: دکتر رضایی

دانشگاه آزاد اسلامی واحد علوم تحقیقات

پاییز و زمستان ۹۵

Table of Contents

۳.....	۱. معرفی نرم افزار ارائه برنامه تبلیغاتی:
۵.....	۲. نیازمندی های نرم افزار
۵.....	۱. ۲، نیازمندی های وظیفه ای:
۶.....	۲. ۲، نیازمندی های غیر وظیفه مندی (non-Functional)
۸.....	۳. معماری سیستم
۸.....	۱. ۲، ذینفعان سیستم:
۸.....	۱. ۳، معماری مورد نیاز نرم افزار
۹.....	۴. ViewPoint های توصیف معماری
۱۰.....	۴، ۲ UseCase View
۱۱.....	۴، ۳ Logical View
۱۴.....	۴، ۴ SubSystems Dependencies With Layers
۱۵.....	۴، ۵ :Implimentation View
۱۶.....	۴، ۶ Deployment View
۱۶.....	۵. نتیجه:

۱. معرفی نرم افزار ارائه برنامه تبلیغاتی:

نرم افزار انتخابی یک نرم افزار پیشنهاد دهنده برنامه های تبلیغاتی می باشد.

این نرم افزار یک نرم افزار -۰۰۰۰۰۰-۰۰۰۰ است.

این نرم افزار اطلاعات کاربران (کارفرماها) را برای تبلیغ در تمامی سیستم های تبلیغاتی برای ایجاد کمپین های تبلیغاتی دریافت می کند و پس از تحلیل و بررسی نیاز های مشتری و با تلفیق با برنامه های زمانی - قیمتی سازمان های تبلیغ کننده (مانند صدا و سیما)، یک خروجی بهینه، مناسب و به موقع به کاربر ارائه می دهد. خروجی ارائه شده باید دقیق، بدون خطا، سریع و دارای زمانبندی و برنامه توزیع بودجه مناسب باشد.

یکی از ویژگی ها و نیازمندی های اساسی این نرم افزار دریافت داده ها و نیاز های مرتبط با خواسته های شرکت های کارفرما (تبلیغ دهنده) است. این اطلاعات باید کاملاً واضح و در قالب های استاندارد دریافت شود تا بتوان از روی آن ها و بدون نیاز به حضور کارفرما تمامی نیاز ها اعم از حوزه کسب و کاری، بازار هدف، مشتریان هدف، رقبای بالفعل و رقبای بالقوه، بودجه تبلیغاتی کارفرما و... را به خوبی استخراج کرد.

تحلیل و تنظیم برنامه های تبلیغاتی در این نرم افزار باید با استفاده از دریافت ورودی های متفاوت مانند جدول زمانبندی تبلیغاتی صدا و سیما و جدول قیمتی صدا و سیما و همچنین جدول مناسبت های اجتماعی فرهنگی باشد که بتواند بهترین خروجی کمپین تبلیغاتی را برای مشتری ایجاد کند.

حساسیت این نرم افزار در استخراج صحیح خواسته های مشتری بدون کوچکترین خللی، به خاطر هزینه های بالای تبلیغات محیطی و رسانه ای است که ممکن است کوچکترین خطایی، محاسبات و نیاز های مشتری را تغییر دهد و این تغییر ممکن است نارضایتی برای مشتری به وجود بیاورد.

۲. نیازمندی های نرم افزار

۲,۱. نیازمندی های وظیفه ای:

- هر کاربر باید ابتدا اطلاعات فردی را وارد نماید و از طریق شماره موبایل و ارسال پیامک شماره تلفن همراهش را تایید نماید تا حساب کاربری فعال شود.
- پس از ورود کاربر یک سری اطلاعات در مورد نوع کسب و کار کارفرما، اندازه شرکت کارفرما، بودجه تبلیغاتی شرکت و همینطور محیط های تبلیغاتی از جمله شبکه های اجتماعی، تلویزیون ، بیلبرد های شهری و... از کاربر دریافت می شود.
- پس از پر کردن اطلاعات سیستم برنامه ها و یا کمپین های پیشنهادی را پس از چند لحظه باید بتواند تولید کند و نمایش دهد.
- در مرحله بعد کاربر می تواند از شرکت تبلیغاتی سازنده نرم افزار ویا همکاران نرم افزار درخواست انجام کمپین را داشته باشد.
- شرکت تبلیغاتی نیز قابلیت گزارش گیری و بررسی لیست سفارشات را دارد.

۲,۱,۱. نیازمندی های مشتریان

- ایجاد حساب کاربری و تایید آن
- فرم ها و صفحات دریافت نیاز مشتری به صورت User Friendly
- قابلیت انتخاب پلن های تبلیغاتی
- قابلیت سفارش به شرکت تبلیغاتی جهت انجام کمپین ها

۲,۱,۲. نیازمندی های شرکت تبلیغاتی

- قابلیت ارسال و بروزرسانی برنامه های تبلیغاتی (شهری – تلویزیون و...) دازای زمان و قیمت و مکان تبلیغات

- قابلیت گزارش گیری از درخواست های ارسالی و یا کاربران ثبت نامی
- دریافت notification به محض نمایش برنامه تبلیغاتی و انتخاب برنامه تبلیغاتی جهت سفارش

۲.۲. نیازمندی های غیر وظیفه مندی (non-Functional)

- **Availability:** این سیستم نباید دچار خطا شود و یا از کار بیافتد و یا پاسخ نا مناسب و غلط تولید کند.

تاکتیک های استفاده شده برای تأمین دسترس پذیری:

Sanity Checking: از جمله تاکتیک های تشخیص خطا می باشد که با این تاکتیک می توان از اشتباهات و خطاهای کامپوننت های برنامه ریزی که وظیفه تلفیق برنامه های متنوع و تطابق با نیاز های مشتری را دارند، اطلاع یافت. همچنین می توان اشتباهات در خروجی ها را نیز تشخیص داد و به حل مشکل اقدام نمود.

Passive Redundancy (warm spare): از جمله تاکتیک های بازیابی خطا می باشد که در

این تاکتیک در زمان هایی که سیستم بار ترافیکی کمی دارد، حالت ها (States) ی سیستم را ذخیره میکند که می توان در صورت از کار افتادن یک کامپوننت، به حالت های گذشته با از دادن مقدار کمی از اطلاعات یا حالت ها دست یافت. در این سیستم مشتری نباید چند باره برای دریافت برنامه کمپین تلاش کند و شکست بخورد و این ممکن است یک ارزش افزوده که قرار است برای شرکت تبلیغاتی به وجود آد را تحت تأثیر قرار دهد.

State Resynchronization: از این تاکتیک برای همسان سازی حالت سیستم در Warm

Spare استفاده می کنیم.

- **Performance:** این سیستم باید بتواند جواب تحلیل ها و گزارشاتش را از چند منبع استخراج

کرده و با نیاز های مشتری تطبیق بدهد و به سرعت به کاربر نمایش دهد.

تاکتیک های استفاده شده برای تأمین کارایی:

Increase efficiency: با استفاده از کامپوننت های هوشمند می توان کارایی سیستم را در

سرعت ارائه پلن و انتخاب تصمیم برای مشتری، افزایش داد. همچنین می توان با طراحی الگوریتم

های بدون خطا و سریع و سبک برای ارائه برنامه های تبلیغاتی کارایی کامپوننت های اساسی را بالا

برد.

Prioritize events: با اولویت دادن به فعالیت ها و رویداد ها برای محاسبات می توان برای

سیستم زمان خرید، در واقع می توان در ارائه برنامه تبلیغاتی ابتدا محیط های تبلیغاتی را نمایش

داد و مشتری را سرگرم انتخاب محیط ها نمود و پس از آن مرحله محاسبات قیمت و زمان

تخمینی را ارائه نمود، این راه حل هم می تواند کارایی را به این شکل بالا ببرد که هم کامپوننت

های پیچیده تخمین زمان و هزینه را ساده تر نموده و هم می تواند برای محاسبات و فهمیدن

خواسته های مشتری با دقت بیشتر و زمان بیشتری عمل نماییم.

- **Testability:** تولید پاسخ های درست و به موقع در این سیستم از تمام خواسته های ما مقدم تر

است، پس لازم است با این نیازمندی غیر وظیفه ای، بتوانیم از صحت و سلامت سیستم در موارد و

اتفاقات غیر منتظره اطمینان حاصل کنیم.

تاکتیک های استفاده شده برای تأمین قابلیت تست:

Abstract data sources: با این تاکتیک می توانیم داده های ورودی و خروجی را ساده کنیم و

داده هایی که احتمال خطا برایشان زیاد است را تست کنیم تا از صحت سیستم اطمینان حاصل کنیم.

Limit non-determinism: در این روش باید از عدم قطعیت جلوگیری کنیم که با استفاده از کنترل ورودی ها به در UI و استفاده از چک باکس ها و Drop Down ها و پیش بینی خواسته های مشتریان در طراحی رابط کاربری از عدم قطعیت جلوگیری کنیم و قابلیت تست را برای نرم افزار بهتر ایجاد کنیم.

۳. معماری سیستم

۲,۱. ذینفعان سیستم:

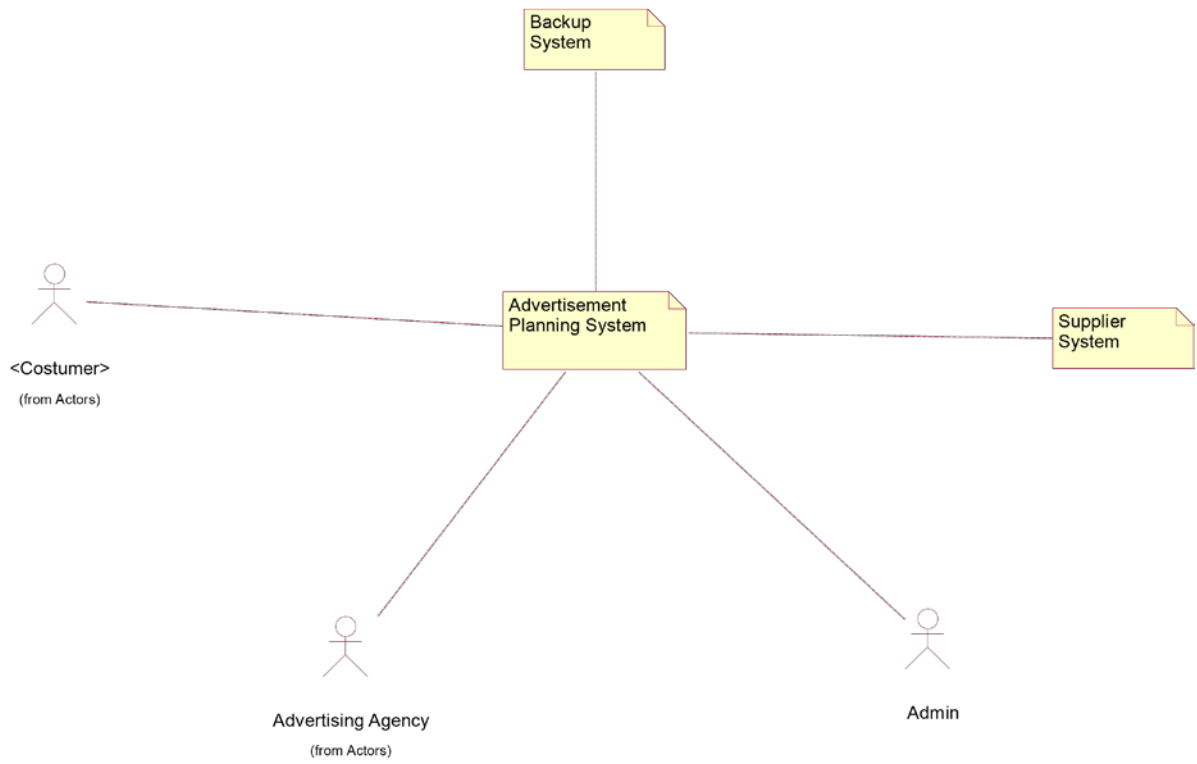
- شخصی که برای محصول یا خدمتی نیاز به بازاریابی و تبلیغات دارد
- شرکت های تبلیغاتی که برای کاهش زمان و هزینه برای برنامه ریزی تبلیغاتی نیاز به این سیستم دارند.
- تیم توسعه برنامه
- کسانی که مسئول آپدیت و اطلاع از قیمت ها و برنامه های تبلیغاتی سازمان های متفاوت مانند صدا و سیما - شهرداری - شبکه های اجتماعی و ... هستند.

۳,۱. معماری مورد نیاز نرم افزار

در این سیستم طبق مطالب کتاب به یک الگوی معماری Module pattern نیاز است. چون این سیستم نیاز دارد وظایفش را تقسیم کند تا در کارایی و قابلیت دسترسی بتواند پاسخگو باشد.

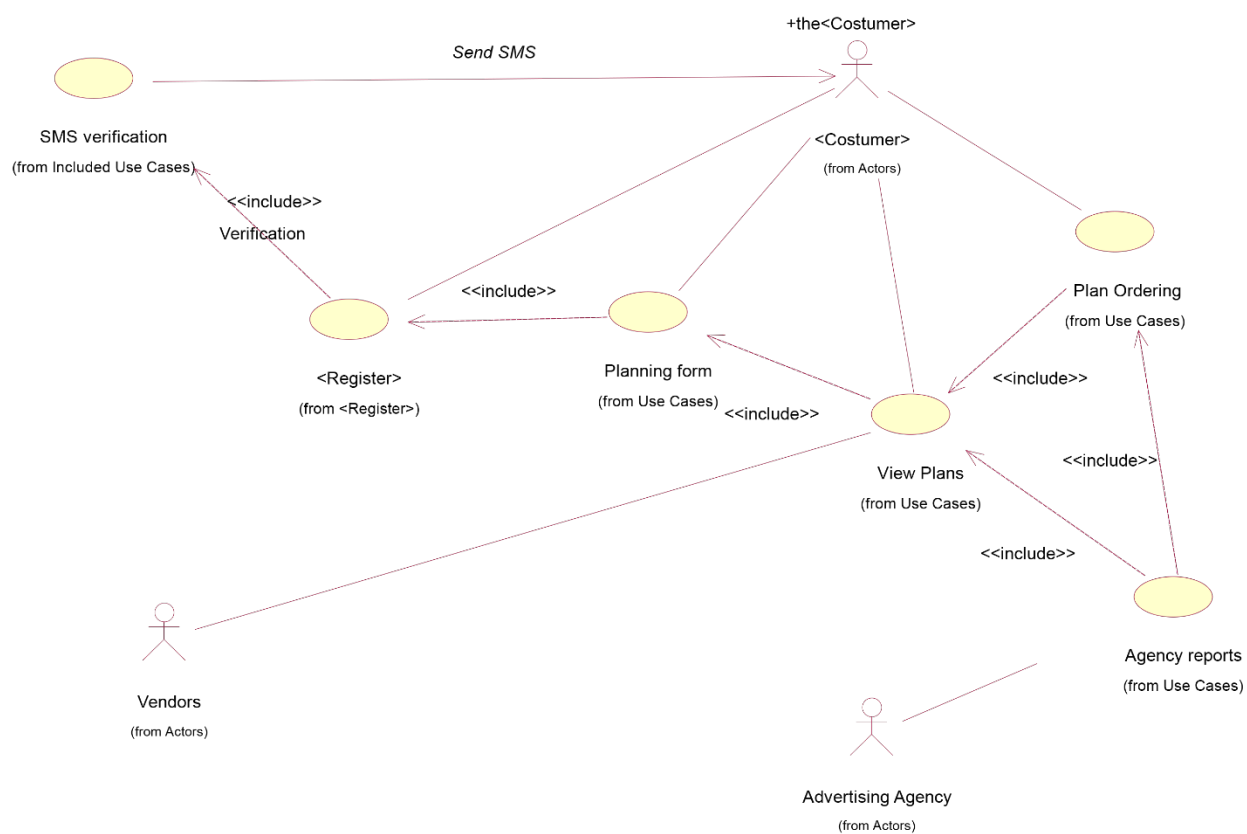
۴. ViewPoint های توصیف معماری

۴.۱. Context View



شکل ۱ - Context View

UseCase View.۴,۲



شکل ۲ Usecase View -

• توضیحات Actor ها:

Vendors: اجاره دهندگان سیستم های تبلیغاتی از جمله صدا و سیما ، شهرداری که برنامه ها و

لیست هزینه - زمان تبلیغات را مشخص می کنند.

Advertising Agency: شرکت های تبلیغاتی که مجری کمپین های تبلیغاتی هستند و برنامه ها را

با توجه به موقعیت شغلی سفارش دهنده تنظیم می کنند.

Customer: مشتریان، شرکت هایی که کالا یا خدماتی برای تبلیغات دارند و نیاز به یک برنامه ریزی

برای موفقیت تبلیغات و بازاریابی محصول خود دارند در عین حال نیاز به اطمینان از صرف درست

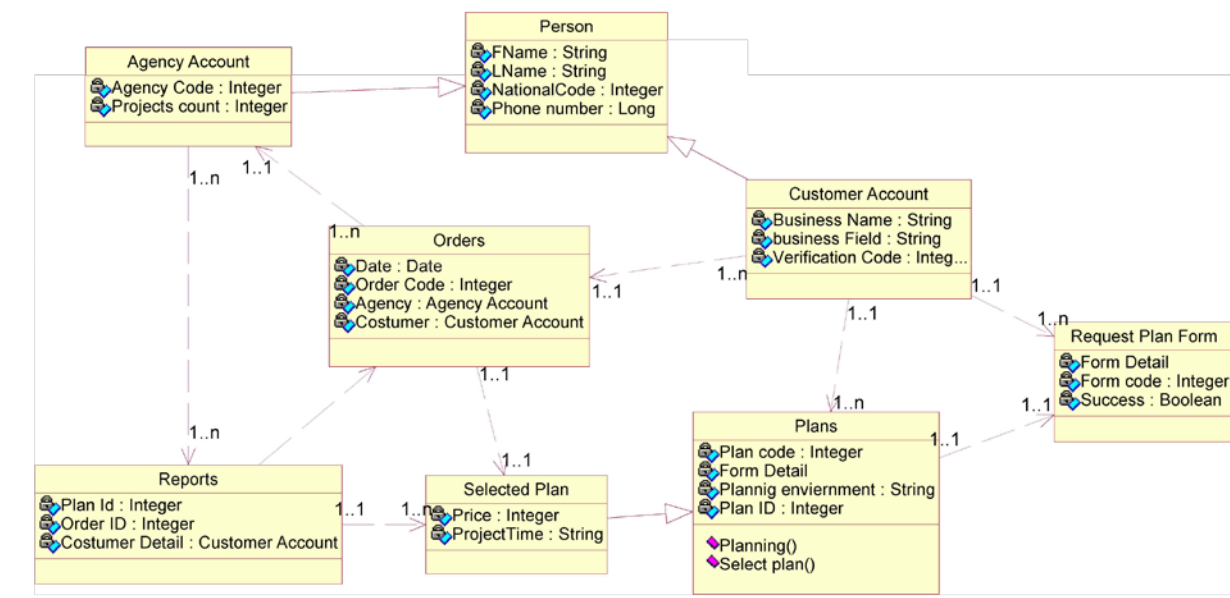
هزینه و زمان خود لازم دارند.

Admin: مسئول رسیدگی و کنترل برنامه های تأمین کنندگان و قرار دادن آنها در سیستم، همچنین

مسئول رسیدگی به سفارشات و درخواست های رسیده از مشتریان است.

۴,۳. Logical View

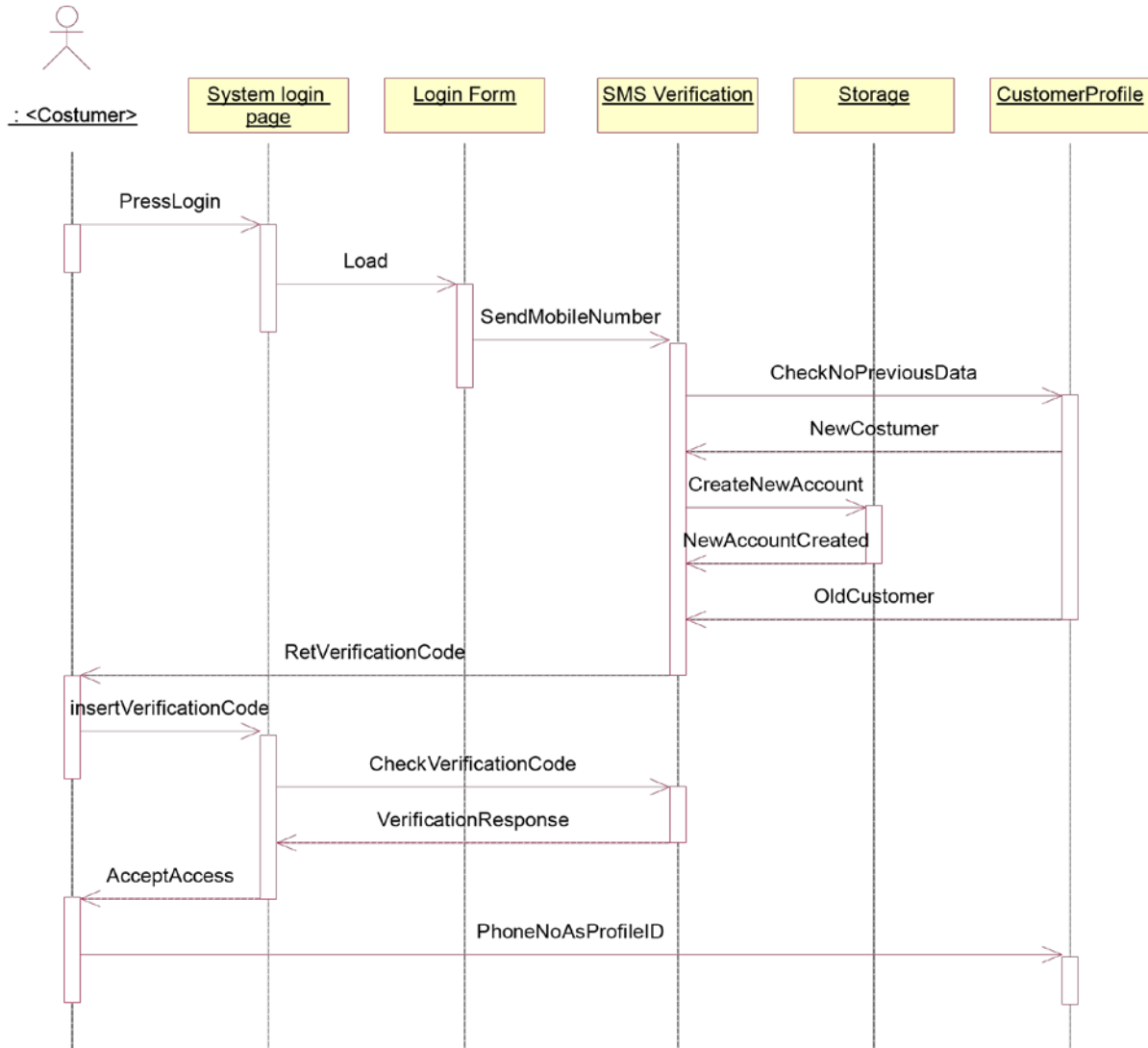
۴,۳,۱. نمودار کلاس:



شکل ۳ یک شمای کلی از قسمتی از نمودار کلاس ها

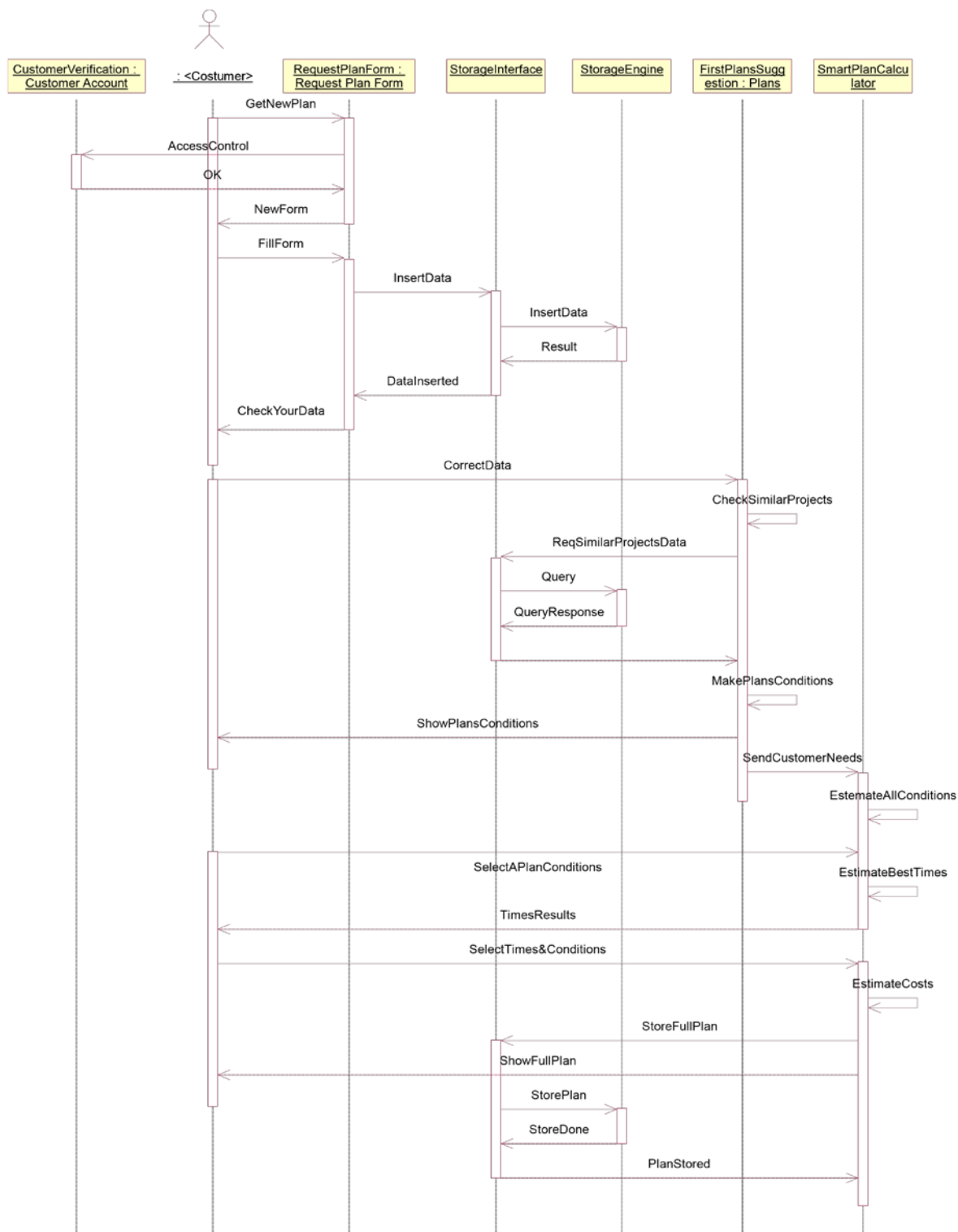
Analysis Interaction View ۴,۳,۲

- نمودار توالی سیستم ورود و اجازه دسترسی کاربر



شکل ۴ نمودار توالی سیستم احراز هویت

• نمودار توالی سیستم پیشنهاد برنامه تبلیغاتی



شکل ۵ - نمودار توالی سیستم ارائه پیشنهاد تبلیغاتی

چگونگی ارضای نیازمندی های غیروظیفه مندی در شکل بالا

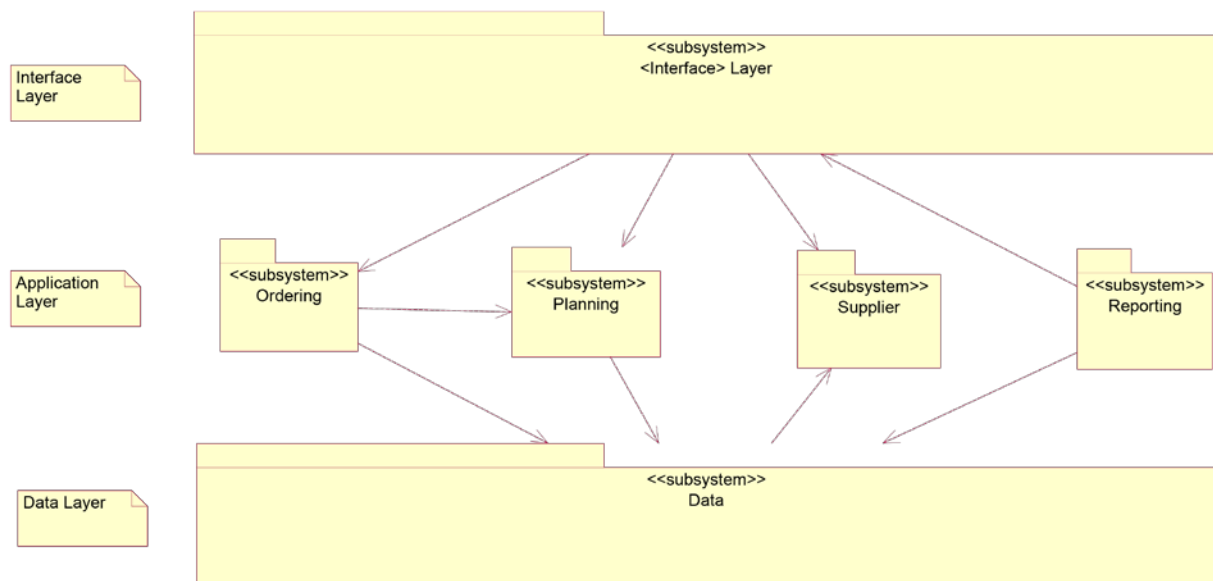
در نمودار بالا برای رسیدن به تاکتیک های کارایی سعی در تغییر بهینه اولویت های سیستم در ارسال درخواست ها و دریافت پیام ها بوده است.

همچنین افزایش کارایی با تقسیم کار بین دو کامپوننت در قسمت تخمین برنامه ارضا شده است.

برای افزایش قابلیت تست از پیچیدگی یک کامپوننت پیچیده جلوگیری شده است.

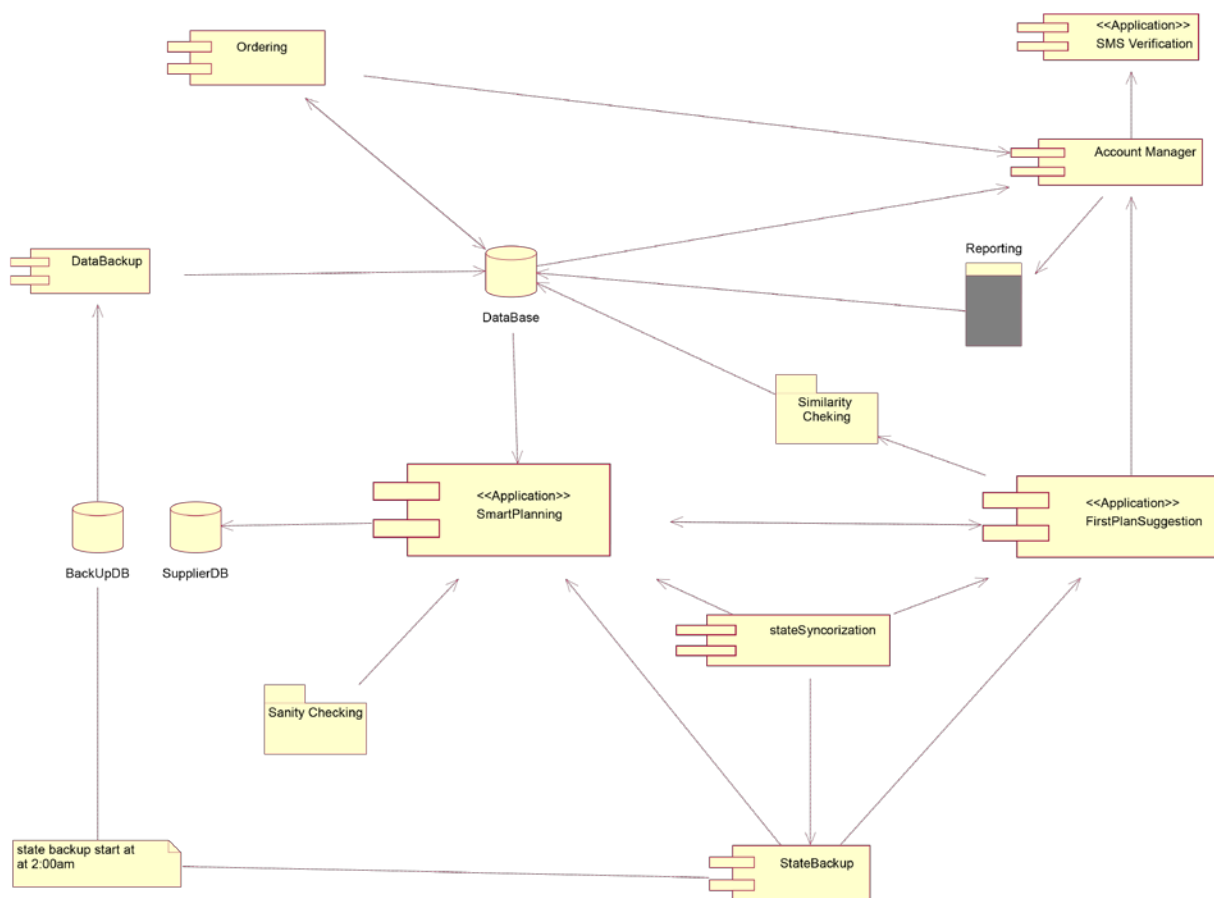
همچنین با ارسال درخواست های پیاپی و بررسی نتایج در شباهت با پیشنهاد های گذشته تست ورودی و خروجی های نرم افزار قابل انجام است.

۴.۴ SubSystems Dependencies With Layers



شکل ۶ - لایه بندی زیر سیستم ها و وابستگی هایشان

۴,۵. Implementation View



شکل ۷ - Component Diagram - Impelementation View

در این View و در کامپوننت های موجود در سیستم برای دسترس پذیری سیستم تدابیری داشتیم.


یکی از این تدابیر Warm Spare در زمان ذخیره داده ها و در زمان پردازش برنامه است. این نرم افزار با توجه

به وابستگی هایی که به داده های بیرونی، داده های پردازش های قبلی، همچنین پیچیدگی در محاسبات و

تخمین دارد نیاز به روشی برای بازیابی دارد.

همچنین با توجه به حساسیت تولید مقادیر صحیح از کامپوننتی جهت Sanity Checking استفاده نمودیم که بتوانیم خروجی های کاپوننت های در حال اجرا را بررسی کنیم و تشخیص خطا را به سرعت متوجه شویم.

۴,۶. Deployment View

با توجه به اینکه این نرم افزار یک نرم افزار تحت وب است نیازی به رسم نمودار لایه فیزیکی نیست و هرآنچه لازم است در نمودار  نمایش داده شده است.

۵. نتیجه:

با توجه به View های به دست آمده معماری لایه ای می تواند معماری مناسبی باشد البته با توجه به نیاز سرعت و کارایی بالا شاید معماری لایه ای اندکی از سرعت ما بکاهد اما از لحاظ Availability و reliability می تواند کارساز باشد.

معماری Client-Server هم می تواند نیاز نرم افزار ما را به خوبی پاسخ دهد و می توان با تقسیم پردازش ها در کلاینت و سرور ها، می توان از بار ترافیکی یک سرور centralized کاهش داد و سرعت و کارایی را بالا تر برد.