۲- بررسی کنید که آیا Class diagram ارائه شده در بخش Structural Modeling نیازمند بهینه سازی می باشد دلایل خود را بیان کنید.

خیر نیازمند بهیه سازی نمی باشد زیرا کلاس های ما کاملا منطبق با اصول SOLID طراحی شده اند و زین سبب نیازمند ویرایش نیستند.

هرکلاس ماصرفاً یک وظیفه بیشتر نداشته و کلاس‌ها فقط و فقط به خاطر ایجاد تغییر در وظیفه‌ای که انجام می‌دهند دستخوش تغییر خواهند شد.

هر کلاس ما برای توسعه یافتن قابلیت‌هایش Open بوده و دست برنامه‌نویس برای افزودن فیچرهای جدید به آن باز می باشد اما اگر برنامه نویس خواست تا تغییری در کلاس ایجاد کند، امکان آن Closed است و او اجازه چنین کاری را ندارد.

کلاس‌های فرزند در صورت موجود شدن آن‌قدر کامل و جامع از کلاس والد خود ارث‌بری می‌کنند که به سادگی بتوان همان رفتاری که با کلاس والد می‌کنیم را با آن‌ها داشته باشیم.

به سادگی می‌توانیم از هر اینترفیسی که نیاز داشته باشیم در کلاس‌های مد نظر خود استفاده نماییم و در صورتی هم که، کلاسی وجود داشت که نیاز به استفاده از چندین اینترفیس مختلف داشت، دست ما باز خواهد بود تا آن کلاس را از چندین اینترفیس پیاده سازی کنیم.

همچنین وابستگی مابین کلاس‌ها، ماژول‌ها و آبجکت‌های سطح بالا با ماژول‌های سطح پایین به حداقل ممکن رسیده است.

۳- خصوصیات coupling و cohesion را برای کلاس های طراحی شده بررسی کنید.

کلاس های ما با توجه به این دو اصول بنا شده اند. البته به صورت مطلق از آنها پیروی نمی کنند.

بر اساس اصل coupling کلاس ها می بایست کم ترین نیازمندی ارتباطی را به یک دیگر داشته باشند. در حالت ایده آل بدین شکل است که کلاس ها اصلا به همدیگر وابسته نباشند. ما این اصل را سرلوحه ی طراحی کلاس های خود قرار دادیم اما برای برخی از قسمت ها مثل بخش E-Wallet و Score Page لازم است آبجکت هایی از هر کدام برای استفاده ی دیگری ایجاد شود.

اصل cohesion میزان انسجام درونی اشیا کلاس ها را بیان می کند. و می توان گفت از این منظر کلاس های ما کاملا بر این اصل منطبق هستند. هر کلاس وظیفه ی انجام یک عملیات خاص را دارد و پیوستگی اشیا برای انجام یک عملیات خاص کاملا مشهود است.

۴- از چه فرمتی برای مدیریت داده ها استفاده می کنید دلایل خود را بیان و تحلیل کنید:

از دیتابیس Relational استفاده می کنیم زیرا علاوه بر مسائل زیر Transaction های زیادی لازم است.
همچنین برای انجام کارها می توانیم از توابع، تریگر، ویو ها نیز استفاده کنیم.
دیتابیس مورد استفاده ی ما mysql می باشد که یکی از بهترین دیتابیس ها برای ارتباط با نرم افزار اندروید است. سمت سرور به راحتی با استفاده از زبان php و از طریق پنل مدیریتی phpMyAdmin به سادگی می تواند با دیتابیس ارتباط برقرار کند و اقدام به ساخت و طراحی جدول ها نماید. این دیتابیس بر روی یک هاست لینکوسی پیاده سازی می شود که از نظر ارتباط و هماهنگی بهترین گزینه برای mysql می باشد. با استفاده از پنل مدیریتی CPanel که قابلیت بررسی امنیت هاست کنترل phpMyAdmin و ساخت API های لازم را به ما می دهد لزوم استفاده از هاست لینکوسی بیشتر احساس می شود.
از دیگر ویژگی های آن portable بودن آن می باشد.
نیاز به سرعت در پروژه ی ما بسیار حیاتی می باشد بگونه ای که لازم است تا کاربر بتواند در هر لحظه و خیلی سریع موقعیت خود و یا وسیله ی نقلیه ی مورد نظر را بر روی نقشه مشاهده کند. این نیاز با این سیستم به خوبی پوشانده می شود.
از آنجایی که این سرور ها همواره آنلاین هستند لذا دسترسی به دیتابیس نرم افزار و سیستم مدیریتی آن همواره برقرار می باشد و reliable بودن سیستم تامین می شود.
به علت حساسیت اطلاعات امنیت بسیار مهم می باشد. با امکانات دیتابیس های رابطه ای همچون view ها و یا سطح دسترسی می توان به راحتی امنیت اطلاعات کاربران و پروژه ها را تضمین نمود. همچنین امکان encryption نیز وجود دارد.

۶- به نظر شما از چه نوع معماری برای طراحی سیستم باید استفاده شود؟

معماری کلاینت سرور بهترین گزینه برای پیاده سازی نرم افزار ما می باشد.
باتوجه این امر که برخی از ویژگی های نرم افزار ما برای هر کاربر منحصر به فرد می باشد و همچنین نیازمند پردازش هایی است که لازم است از سوی کاربر صورت گیرد معماری کلاینت سرور پیشنهاد می شود. برای مثال پردازشی همچون پیدا کردن موقعیت کاربر می بایست با استفاده از GPS مربوط به گوشی هر کاربر مورد بررسی قرار گیرد.
از طرفی با توجه به حجم عظیم محاسبات و پردازش های لازم برای تحلیل بهترین مسیر و رهگیری مسیر بهترین گزینه آن است که این فرآیند توسط سرور صورت پذیرد. نیاز به انجام پردازش سریع مسیر و نمایش لحظه ی موقعیت وسیله ی نقلیه و همچنین کاربر از دیگر عللی است که لازم است که حضور یک سرور قدرتمند را الزامی می کند.
لایه های presentation و همچنین بخش هایی از لایه ی Applications بر روی کلاینت و سایر لایه ها در سمت سرور پیاده سازی می شود.

معماری کلاینت سرور مزایای زیر را برای پروژه به ارمغان می آورد:

1) هزینه ی پیاده سازی این معماری نسبتا پایین بوده و به راحتی می توان زیر ساخت آنرا ارتقا داد

2) با توجه به نیاز به کنترل بهیه ی نرم افزار و حفظ امنیت آن معماری کلاینت سرور می تواند این نیاز را پوشش دهد.

4) مدل کلاینت سرور برای گسترش GUI و امکانات ظاهری مفید واقع می شود و از آنجا که کاربر با سیستم کار می کند باید ظاهر مناسبی به او ارائه گردد.

5) مدل کلاینت سرور برای development می تواند مفید باشد. چون تغییرات در برنامه سرور منجر به تغغیرات کلان در برای کلاینت نخواهد شد.