1. 你的设计支持语言转换吗？（请举例） 这个部分有bonus，还是想加上去一下，但是只需要做accueil和main page的语言转换就可以了，别的地方先不做，只是给老师举个例子可以转换语言
2. 评估一下这个设计从网页切换到手机的难度 请说明

这个难度不知道怎么怎么描述，要切换到手机app，举个例子要在安卓手机上做一个app，可以做原生安卓项目，也可以做html5的通用项目，前者要把所有代码都重写，非常费时费力，并且该做法也不够通用，目前大部分的移动端开发都不会采用这种策略。后者现在应用比较广泛，并且使用类似于cordova这样的框架，可以同时导出安卓和苹果的应用，并且做项目移植，前端的页面元素几乎不用改动，只要改动相应的css，使其能自动适应屏幕宽度就行，后端的逻辑也不需要在手机端实现，可以把网页端的后端做成接口，同时被手机端和网页端调用就可以，总得来说要实现手机和网页同时可用，就需要把目前项目的前后端分离，后端项目用flask框架，前端只存网页及页面交互的js和css样式

1. 模块化是决定architecture的质量的一个重要标准（一个组件（composant）或一个class只有一个目标）你的体系结构的组件是否遵守此模块化的原则？请证明

基本符合的，因为python项目跟java项目有一些区别，没有明显的class区分，但是对于后端逻辑来说，django框架中，web\_historien是项目名称，这个目录下有整个项目通用的settings相关的目录（web\_historien），所有页面html目录（template），静态文件例如图片js和css（static）目录。

app01则是对于业务处理的后端代码目录。在app01里，views通常是用来处理所有页面需要的业务逻辑，为了便于开发并且遵循模块化原则，在代码中把搜索，预约和需求三个模块的逻辑分别拆分在3个views.py文件里，分别是views.py，views\_search.py， views\_demand.py，而对于通用的每一个业务方法都需要做的事，比如验证用户登录的功能，或者获取通用的页面下拉框列表等，则提取到了utils.py 模块中。

models.py是所有与数据库对应的模型类，一个模型对应一个类，这显然是遵循的模块化的原则。

1. 你的架构元素（element of architecture）可以重新应用到另外的项目中？请说明

Django框架是一个通用的web应用框架，可以非常随意的应用到任何其他需要轻量级MVC架构的web应用中，按照相应的目录结构配置，只要编写前端页面，配置页面元素的事件需要对应后端哪个url的方法来处理，并在相应的处理方法里写业务逻辑，最后以render或者JsonResponse的方式返回给前端整个渲染好的页面或者Json数据就可以了。

1. 你的architecture允许你轻松改变他的数据库管理系统（SGBD）吗？ 请说明

允许的，

首先在settings.py里可以配置一个或者多个数据库的连接，业务代码里可以控制哪段逻辑需要哪个数据库连接，

models.py文件定义的数据类型都是django框架提供的通用数据类型，不会因为连接的数据库不同，而需要重写这个模型类的定义。比如一个autoField，如果用了mysql数据库，django在把model migrate到数据库时，就会创建一个int类型的字段，并且设置成auto increment，如果用的是sqlite3数据库时，django也会创建一个Integer类型的字段，并设置成AUTOINCREMENT。

最后编写完models文件的数据定义后，可以轻松的使用makemigrations 和 migrate命令，把models文件定义的数据映射到数据库中，这两个python命令包含了所有数据库中创建表，添加主键外间关系的命令，用户不用关心具体的数据库命令的区别。

1. 如果你想要完全改变GUI的设计，存不存在某些composant或者class是可以保持不变的？

GUI是用前端语言编写的，不存在class文件，更不用说能否保持不变。

目前项目中对于前端来说，页面布局都是写在html里的，页面元素的样式和排版都在css文件中，而跟后端交互的页面元素的事件(比如点击事件，下拉框选择事件，单选复选框选择事件等)都写在js里。如果要改变GUI的设计，那么只有js里的内容可以不用改变，html里的元素和css里的样式肯定需要相应改变的。

1. 如果更改了数据库的结构（添加新表或属性，更改表或属性的名称，更改属性类型，更改约束。。。）可能会产生哪些影响？请说明

第5题说过了，编写完models文件的数据定义后，可以轻松的使用makemigrations 和 migrate命令，把models文件定义的数据映射到数据库中，这两个python命令包含了所有数据库中创建表，添加主键外间关系的命令，用户不用关心具体的数据库命令的区别。业务逻辑代码中，也没有一句sql语句，都是用的面向对象的思想，从models中获得相应的class，读写class的属性，django框架在运行时会自动帮你转换成对应的sql语句。

唯一值得注意的就是，每次想要更改数据库的结构，都要重新运行makemigrations和migrate命令，把最近新的结构变化同步到数据库中，而业务逻辑的相关字段的读写也要做相应的改动。

我怕我有些翻译的不对，下面我把法语原版的放上来 ：

1. Votre architecture vous permettrait-t-elle de supporter facilement une autre langue utilisateur (Français et Anglais par exemple)?
2. Evaluez la difficulté du passage de votre application web à du mobile.
3. La modularité est un critère important de qualité des architectures (Un composant ou classe n’a qu’un seul objectif). Les composants de votre architecture respectent-ils ce principe de modularité?
4. Des éléments de votre architecture seraient-ils réutilisables dans un autre projet?
5. Votre architecture permettrait-elle de change de SGBD facilement?
6. Si vous deviez changer totalement la réalisation de l’interface graphique, y’aurait-il des composants ou classes inchangé(e)s?
7. Quels seraient les effets d’un éventuel changement dans le schéma de la base de données (ajout d’une nouvelle table ou nouvel attribut, changement du nom d’une table ou d’un attribut, changement sur un type d’attribut, sur une contrainte…) dans l’application?