本作品名Habit，中文意思是习惯。

本作品的灵感来源于Keep，众所周知Keep是个健身app，它的核心功能就是设定并keep it，我在这基础上，将作用对象从健身扩展到了习惯。人们可以使用本应用来养成习惯，设定一个习惯后，就可以在习惯页面上看到自己设定是各个习惯，还能看到坚持了几天，时刻提醒自己要坚持下去。除了能看到习惯以外，人们还能在心得页记录心得，记录在习惯养成路上的点滴心得，在养成习惯时也收获成长，提醒自己不忘初心。

获得作品创意后，便开始着手作品的设计。首先是整个项目的架构、结构设计。由于我们相对于MVVM（model view viewmodel）的设计模式来说更熟悉MVC（model view controller）的设计模式，故我们选择了以NodeJs的Express框架来作为整个项目的框架。数据库的连接上我们选择了MongoDB，一种非关系型数据库（nosql），也是近年来网站连接数据库的热门类型，因为nosql具有应对高并发的优势。既然选择的是MongoDB，我们的model模型层的结构由schema-model构成，schema对象保存数据类型、关系等，由model将其抽象，而具体的实例化则是document；view层则是前端页面，分为main、admin两部分，也分别由两个对应的路由文件负责转发；controller层则是router和api的合二为一，负责路由转发、业务逻辑，model层从view接收到的业务逻辑将在controller层解决。

完成项目的结构设计，接下来是对view层——前端页面的设计。我们将view层分为main、admin，对应有路由文件main.js和admin.js。我参考了下其他网站，发现都有个门面页面，用户在门面页面能看到网站介绍并且跳转的登录注册页面，因此我们的main分成了index首页和login登录页。首页制作很简单，考虑到最近流行的至简主义，分为跳转到登录和应用简介两部分；登录页由背景和登录部分，背景引用的是GitHub上的开源项目的文件，一种制造3d粒子的js文件，登录注册部分做成一张卡的样子，点击按钮可以在登录注册切换。登录后用户进入后台，后台我设计成三部分：个人中心页、记录心得页、Habit页。考虑到用户想快速跳转其他功能，每页都有个通用的nav导航栏，可以轻松在各个功能之间跳转。这部分的实现都在路由的js文件里实现，十分简单。而用户想要创建、编辑删除心得或者习惯则需要连接MongoDB，通过路由或者api文件来实现相关功能。

做完pc端后要考虑的是mobile端的用户体验。一开始我想的是流式布局，让页面自适应，也能少写很多代码。后来我发现mobile端体验真是糟糕的不行，开始考虑真正实现响应式布局。响应式布局主要有两套方案，一套是用js来控制变量，也就是观察者模式，这种模式实现的比较好，但是相对应的代码也会多很多，难度那也是大幅度提高；另一套方案是写两套布局，也就是用media query，针对device-width来控制css，好处是写的快，简单，但是不能面面俱到。考虑了一下下，我就选择了media query，毕竟时间好像不是很多，来不及精确到每个人，而且实际也没太多差距。敲定了方案后，我就针对mobile端页面重新设计了一套新的页面，因为pc端页面好像不太适合手机端的设计习惯。同时我还在手机端加了个抽屉功能，考虑到手机应用都有的功能，便做了一个左边拉出的抽屉。

做完了所有该是项目的部署。我最开始尝试在大赛指定的时速云上部署，奈何登不进去。刚好我自己也有阿里云的ecs云服务器，并且拥有一个域名，就决定部署在自己的阿里云云服务器上。将项目托管到GitHub上后，我在服务器上通过git clone下载了项目。Web服务器我直接用的NodeJs，代理服务器我选择的是Nginx。由于我买的是华东2的服务器，考虑到北方会有延迟，我选择在Nginx托管项目的静态文件。将DNS解析到域名后，我在Nginx上设置了转发到对应的端口。到这里设置完毕，就可以通过域名访问了。

最后是测试与优化部分，一部分的测试用例在项目进行时就已经测试完毕，还有一部分由于我们都不是专业的测试人员，便考虑灰度测试，找人来试验网站，提供意见、找出bug。找了一些朋友体验测试后，确实修改了一堆bug并优化了一部分（10分钟内改了7处）。直到现在的版本，算是没什么大问题，也能进行正常使用。

作品还有很多想要开发的功能，但是考虑到时间，而且核心功能也已经做完并无什么问题，就准备在以后慢慢的添加新的功能，也就是更新与维护啦。