Case-B 项目文档-User1

组别：工具组

项目文档

# Introduction

### 软件需求规约说明书 - 第一章：Introduction  
  
#### 背景  
在当下信息化发展迅速的时代背景下，城市的交通体系变得日益复杂，传统的导航方式已经无法满足当前人们对时间效率、出行体验的需求。城市中的通勤者每天面临多种交通方法和多变的路况信息，使得导航出行工具的有效性和准确性变得更加重要。在这种背景下，City Mapper——一款交通导航应用应运而生。City Mapper凭借其强大的功能和精准的数据处理能力，旨在提供更加高效便捷的出行解决方案，帮助用户快速找到最优的出行路线。  
  
City Mapper结合先进的技术和数据分析手段，不断优化其系统功能，以确保无论是日常通勤还是旅游探索，都能为用户提供方便快捷的服务。已经开发的City Mapper在上线前，基于大量实际应用案例和用户反馈，不断完善其内部功能模块，实现系统的高效运行，从而满足不同用户群体的需求。City Mapper软件致力于通过提供详尽的导航服务、丰富的兴趣点信息以及实时的交通更新，成为用户最值得信赖的导航助手。  
  
#### 业务目标  
本文档旨在明确City Mapper的软件需求，为开发团队提供清晰的指导和方向。通过定义详尽的功能、非功能需求及相关设计边界，确保软件开发过程中的所有参与者能够达成共识，并根据这些目标进行产品设计、开发、测试和维护工作。本文档的编制基于City Mapper的研发背景和实际应用场景，力求在准确传达软件功能需求的同时，也能够适应市场和技术发展的动态变化。  
  
#### 范围  
本文档涵盖City Mapper的所有功能需求、非功能需求以及相关的约束条件。包括但不限于用户界面设计、用户交互流程、后端数据处理逻辑、安全性设计及系统可扩展性要求等方面。具体范围包括：  
  
- 功能需求部分将全面描述City Mapper的各项功能特点，如创建路线、查询路线、更新路线信息、删除路线、管理交通方式、查看实时交通状况、更新公交时刻表、添加和删除兴趣点、用户注册与登录、用户资料管理、添加和管理距离等。  
- 非功能需求部分涵盖了性能指标、用户界面设计原则、系统的安全性、可用性、可维护性和可扩展性等要求。  
- 约束条件指出了开发过程中必须遵循的行业标准及相关法律法规，以及软件设计和应用场景中的特殊限制要求。  
  
#### 定义、缩写与缩略语  
在该文档中，您将遇到一系列术语和缩写，它们指的是特定的技术工具、系统功能或发展概念，需要明确定义以避免术语含糊不清或理解不一致。以下为本文档所使用的部分重要术语定义及缩写说明：  
  
1. \*\*City Mapper\*\*: City Mapper是一款专注于城市交通导航的智能应用，其功能包括但不限于路线规划、交通状况实时更新、兴趣点信息提供等。  
2. \*\*Transport\*\*: 交通工具是城市居民出行方式的载体，包括但不限于公交车、地铁、步行、骑行等。  
3. \*\*Route\*\*: 路线指用户出行时规划的从起点到终点的路径，可以包含不同的交通工具和兴趣点。  
4. \*\*Real-Time Traffic Conditions\*\*: 实时交通状况实时更新交通情况，为用户提供实时、动态的交通信息反馈，能够帮助用户提前了解路途中可能出现的各种问题。  
5. \*\*Public Transportation Schedule\*\*: 公共交通时刻表提供的时刻信息，基于特定交通工具，提供运营时间和频率，确保用户准时出行。  
6. \*\*Points of Interest(POI)\*\*: 兴趣点包含用户感兴趣的城市热点，如餐厅、博物馆、地标建筑等。  
7. \*\*RouteManagementRecord\*\*: 路线管理记录记录每一路线的历史操作，确保用户可以追溯每一个操作进度。  
8. \*\*RoutePreferences\*\*: 路线偏好用户根据自己的需求或喜好设定路线类型，如偏好快速或距离短。  
9. \*\*User Feedback\*\*: 用户反馈为用户提供提交反馈的途径，确保用户使用体验的有效记录。  
10. \*\*DistanceManagement\*\*: 距离管理计算两点之间的距离，并在系统中记录相关属性值。  
  
#### 参考标准  
本文档的编写基于以下文档、标准和其他参考资料：  
  
- \*\*行业标准\*\*: 包括但不限于ISO/IEC 25010:2011《软件工程——产品和系统质量要求和评价（SQuaRE）—系统和软件质量模型》，CIW（Certified Internet Web Professional）的知识模块，各类UI设计指导原则等。  
- \*\*法律法规\*\*: 根据开发地区和目标市场的实际情况，可能会引用到《个人信息保护法律法规》、《互联网信息服务管理办法》等，确保软件产品满足相关法律要求。  
- \*\*用户研究与市场调研\*\*: 用户研究与市场调研报告，包括但不限于竞品分析、用户访谈、市场调研及用户反馈等，都为本文档提供了宝贵的信息来源。  
- \*\*开发与技术文档\*\*: 涵盖Branch Source/MSDN文档等在线资源，也包括在开发过程中形成的内部文档、代码标准和开发指南等。  
  
本项目通过定义全面的需求规约说明书，从结构上引导并约束功能性开发，以此确保软件开发过程中所有参与者包括开发团队、产品团队、项目管理团队共识一致且有序地推进，最终高效产出可满足需求的高质量市民通勤助手——City Mapper。  
  
---  
  
以上文档全面阐释了City Mapper软件需求规约说明书第一章Introduction的主要内容。引言部分涵盖了项目背景、解决方案、目标受众、开发原因以及重要性等关键内容，为我们提供了较为详尽的项目起点，保障后续部分的可读及统一性，更进一步确保整个开发团队清楚的了解为何及如何构建如此高品质的应用系统。

# Overall Description

### 总体概述  
  
#### 产品视角  
  
\*\*愿景\*\*：在现代都市的快节奏生活中，高效、便捷的路线规划成为人们出行的重要需求。我们的软件产品——城市路径导航器（City Navigator）的目标是通过将先进的交通技术和人工智能算法融合在一起，为用户提供简单、快捷、实时的出行导航服务，确保每一位用户在任何时间、任何地点都获得最优的路线规划和交通建议。  
  
\*\*目标用户群体\*\*：我们的目标用户群体广泛，包括但不限于日常通勤族、旅游者、商务人士、学生等。不论是在繁忙的城市中穿梭，还是在外地旅行，City Navigator都能为用户提供最佳的出行解决方案。  
  
\*\*预期市场定位\*\*：希望将City Navigator定位为具有高可用性、可靠性和智能化的领先都市交通辅助软件。通过集成丰富的交通信息资源和独特的用户界面设计，给用户提供优秀的用户体验，实现满足各类需求和广泛的市场份额。  
  
#### 产品功能  
  
\*\*概述\*\*：City Navigator是一个整合路线规划、实时交通信息、公共交通时刻表、兴趣点管理以及用户资料管理的综合服务平台。  
  
- \*\*路线规划\*\*：用户可以创建自定义路线，查询已存路线，更新和删除路线。此外，系统提供多条路线选择，支持用户根据实时交通状况和不同的兴趣点偏好选择最佳路径。  
- \*\*实时交通状况\*\*：用户通过查看实时交通信息了解道路拥堵情况，以便做出明智的行驶决策。  
- \*\*公共交通时刻表\*\*：包含公交、地铁等公共交通工具的运行时间，便于用户合理规划行程。  
- \*\*兴趣点管理\*\*：允许用户添加、编辑个人兴趣点（如餐厅、景点等），以便于在规划路线时将其考虑在内。  
- \*\*用户资料管理\*\*：用户可以管理个人信息，包括姓名、电子邮箱、密码等，并设置个人偏好，如选择偏好的交通方式等。  
  
#### 用户特征  
  
我们的目标用户群体广泛，包括日常通勤、旅行和商务活动中的各年龄段人群。依据用户的使用习惯可以分为以下几类：  
  
- \*\*技术熟练度\*\*：大部分用户具有中等程度以上的智能手机使用技能，能够独立操作基本功能。对APP的深入使用存在一定差异，一部分可能更倾向于使用更加直观、便捷的功能。  
- \*\*使用频率\*\*：包括常见的日常上班族和商务人士，他们每周可能使用多次；包括旅游爱好者，他们可能会在旅游期间频繁使用此类APP。  
- \*\*操作偏好\*\*：一部分用户希望系统自动进行路线规划，而另一些用户则更喜欢手动选择特定路径。用户也希望通过不同的方式对比现实状况，以更好的评估和选择合适路径。  
- \*\*人性化设计\*\*：用户期待在计划行程时能够有合理的视觉和导航设计，使得操作流程简单明了，方便用户快速反馈。  
  
#### 竞品分析  
  
\*\*市场痛点\*\*：  
- 大型城市的公共交通系统复杂多样，难以提供详尽时刻，造成用户的出行困难。  
- 找不到路径规划的高效工具，缺少实时交通信息和兴趣点推荐建议。  
- 现有交通导航软件功能单一，未能将多种信息融合在同一体系中，需要频繁在不同软件之间切换，操作不够流畅，用户体验一般。  
  
\*\*同类软件对比\*\*：  
- 大部分竞品如Google Maps和Apple Maps等，提供了类似路线规划、交通状况查询的功能，但这些平台偏向于简单的路线导航，缺乏全面的后续路线管理和服务。  
- 很多导航应用在生成路线后不再提供更多的服务，对于实时交通变化和出行兴趣需求的满足程度不高。  
- 竞品中鲜少有完整的从用户创建路线到管理路线兴趣点的闭环服务系统，因此City Navigator致力于解决以上问题，为广大用户提供更高效和人性化的出行体验。  
  
\*\*竞争优势\*\*：  
- 整合交通方式管理和实时交通情况显示，帮助用户实时掌握并调整路线规划。  
- 提供公交时刻表更新功能，用户可以更为精准地规划公共交通的使用。  
- 统一平台内的兴趣点添加与管理功能强化了个性化的路线推荐，增加系统粘性。  
- 用户可以享受到从注册、登录到个人信息、路线记录等方面的全能管理，避免信息分散用户难以自行整合的问题。  
- 数据安全及隐私保护措施到位，充分保障用户的个人隐私和数据安全。  
  
#### 业务风险与机遇  
  
\*\*业务风险\*\*：截至当前，可预见的业务风险包括市场接受度的风险，即未来用户是否会广泛接受该软件并愿意使用它规划路线以及实时查询交通状况；面临新技术的快速迭代，在不断变化的科技领域中我们需要应对技术更新带来的挑战；以及对于提供处理海量用户的实时交通数据和准确模式匹配能力的要求，需要更高的IT系统准确性与流利的硬件适配技术能力。  
  
\*\*业务机遇\*\*：City Navigator正把握一系列趋势，包括但不限于数字化时代的出行偏好，随着城市化进程的不断推进和科技的发展，大众出行对智能导航的需求将不断增加；City Navigator能够通过精准预测与整合资源在激烈的交通软件市场竞争中迅速聚集用户；而通过定向推广目标人群，以及与当地政府和各类公共事业部门的合作，我们能够通过良好信息互通，积极构建共赢生态体系进一步扩大用户使用场景。  
  
#### 设计约束  
  
\*\*技术栈\*\*：我们将使用现代的Web技术和最新的移动应用开发框架，如React Native或Flutter等，确保跨平台兼容性、高性能和用户体验的流畅性。还需与多个服务平台之间进行服务集成能力，确保API兼容性及生态系统的一致性。  
  
\*\*硬件平台\*\*：鉴于目标用户群的设备多样化，City Navigator将保持对iOS、Android等多个主流操作系统版本的支持，并注重在不同屏幕尺寸和分辨率上的适配，以适应各类智能手机和平板电脑。  
  
\*\*法规要求\*\*：在涉及隐私数据收集、存储和传输的环节，City Navigator严格遵守用户隐私权保护法案。且软件中可能包含交通信息和其他敏感身份数据，开发团队需特别注重确保遵守相关法律与合规规定，并征得合法授权并保存用户数据最多两年。  
  
#### 假设与依赖关系  
  
\*\*假设\*\*：文档中假设用户能够接受和操作智能交通软件，同样地，预期用户对于订阅和反馈行程的路程数据保持灵活采用。初步的开发进程依赖大规模用户测试结果进行后续调整，在此期间需要确保信息畅通保障稳定性能，避免软件因早期漏洞导致庞大的用户群体出现接受问题。开发团队需持续通过用户体验优先原则，根据用户反馈及市场反馈不断改进产品功能设计。  
  
\*\*依赖关系\*\*：本软件与第三方服务有若干上下游关系，包括但不限于政府提供的实时交通数据源，通过API对接实现同步；第三方服务场所标注，如知名旅游景点、餐馆等信息更新及时交互显示；需要依赖地图基础服务提供完善的地图数据以便定位执行轨迹；还有公交车、地铁的运营时刻等公共交通服务的信息更新集成服务，以供用户实时参考、查询。  
  
以上若干因素都为本产品在市场竞争转化为优势提供了充分的保障。各大重要功能模块清晰，角色业务流程分明，对于产品后期进行维护或新增功能都有较高的可行性和扩展性。希望本产品可以为广大用户提供优秀的出行体验，便捷的出路解决方案，与社会各界伙伴互利共赢，创建开放共赢的生态圈。

# Functional Requirement

以下是根据您的请求，为City Mapper应用生成的完整的功能需求。这些功能需求直接对应于提供的用例描述以及实体关系图中的实体集。  
  
### 功能需求  
  
- \*\*用户注册\*\*  
 - 描述：新用户可以通过输入基本信息来创建账户。  
 - 输入：用户的姓名、电子邮件地址、密码。  
 - 输出：新创建的用户账户，用户可登录到系统。  
   
- \*\*用户登录\*\*  
 - 描述：已注册的用户通过输入姓名和密码验证身份后登录系统。  
 - 输入：有效的用户名及其关联密码。  
 - 输出：登录到系统。  
  
- \*\*用户资料管理\*\*  
 - 描述：允许用户管理个人资料，包括查看及修改基本信息。  
 - 输入：被修改的用户信息，如姓名、密码等。  
 - 输出：用户账户的最新信息。  
  
- \*\*创建路线\*\*  
 - 描述：用户选择并创建从某个起点到终点的路线，路线考虑交通方式和附近的兴趣点。  
 - 输入：起点、终点及所需的交通方式和兴趣点等。  
 - 输出：包含所有路径详情的路线查询结果和系统中的新路线记录。  
  
- \*\*查询路线\*\*  
 - 描述：用户可以查询和查看已创建的某个路线及与之相关的实时交通信息与兴趣点。  
 - 输入：特定路线查询标识。  
 - 输出：根据设置的筛选条件返回的路线信息。  
  
- \*\*更新路线信息\*\*  
 - 描述：允许用户修改已经创建的某条路线的信息，包括交通方式和兴趣点。  
 - 输入：被更新的特定路线的详细信息。  
 - 输出：系统中对应路线的更新记录。  
  
- \*\*删除路线\*\*  
 - 描述：移除用户不想保留或不再需要的路线信息。  
 - 输入：需要删除的路线标识符。  
 - 输出：系统中删除了指定路线的验证信息。  
  
- \*\*管理交通方式\*\*  
 - 描述：用户管理和改变与其路线关联的交通方式。  
 - 输入：选择需要更改的交通方式信息。  
 - 输出：新的或修改后的交通方式及路线信息。  
  
- \*\*查看实时交通状况\*\*  
 - 描述：用户可以查看某一路线的实时交通状况，以指导出行计划。  
 - 输入：查询交通状况的路线。  
 - 输出：系统的实时交通数据报告。  
  
- \*\*更新公交时刻表\*\*  
 - 描述：用户可以添加或修改系统的公交时刻表信息，以反映最新的时刻变化。  
 - 输入：公交线路及更改或添加的时间信息。  
 - 输出：更新后的时刻表数据。  
  
- \*\*添加兴趣点\*\*  
 - 描述：用户可以通过系统添加新的兴趣点，供自己及其他用户查阅。  
 - 输入：新的兴趣点详细信息。  
 - 输出：系统数据库中的新增兴趣点信息。  
  
- \*\*删除兴趣点\*\*  
 - 描述：用户可以删除系统中的兴趣点，前提条件是用户有权管理该兴趣点。  
 - 输入：需要删除的兴趣点标识符。  
 - 输出：从系统中删除的兴趣点信息。  
  
- \*\*添加距离\*\*  
 - 描述：用户能够添加距点至系统，帮助其他用户理解不同地点之间的距离。  
 - 输入：起点、终点及其他必要信息用以计算距离。  
 - 输出：系统中新增的距离记录。  
  
- \*\*管理距离\*\*  
 - 描述：用户可以查看和修改系统中的距离记录，以确保信息的准确和更新性。  
 - 输入：被查看或修改的距离信息。  
 - 输出：系统中更新后的距离信息。  
  
这些功能需求旨在全面覆盖City Mapper应用的主要功能和用户能够执行的操作，并确保所有输入和输出数据都在定义的功能及实体关系内部。

# External Description

### 第四章：External Interface  
  
本章节详细描述了City Mapper软件与外部系统（包括硬件、其他软件和服务）之间的接口需求。这些对接包括各种通信协议、数据格式和接口类型，旨在确保系统的功能完整性和数据交换的准确性。  
  
#### 4.1 硬件接口需求  
  
- \*\*硬件兼容性\*\*: City Navigator旨在支持各个平台设备，包括iOS和Android操作系统下的智能手机和平板电脑，确保在不同品牌和型号的设备上均可正常运行。  
  
- \*\*传感器集成\*\*: 利用设备内置的传感器（如GPS），支持实时定位功能，提供地理位置服务。  
  
- \*\*屏幕分辨率及尺寸适应\*\*: 软件能够适应不同的屏幕分辨率及大小，保证在不同尺寸设备上的用户体验一致性。  
  
- \*\*硬件加速\*\*: 支持硬件加速技术（如GPU加速），以确保地图渲染和动画效果的流畅性。  
  
#### 4.2 软件接口需求  
  
本部分描述了City Mapper软件与第三方软件和服务的接口需求，这些接口包括但不限于API调用、数据交换等。  
  
- \*\*第三方地图服务API\*\*: 基于开源或商业地图服务供应商如Google Maps API、高德地图API等，用于获取地理数据如路线规划、地点搜索和实时地图渲染。  
  
 - \*\*接口类型\*\*: RESTful API, Web API  
 - \*\*通信协议\*\*: HTTP/HTTPS  
 - \*\*数据格式\*\*: JSON  
 - \*\*功能\*\*: 路线规划、地点搜索、地形信息及实时地图更新  
  
- \*\*实时交通状况API\*\*: 连接至实时交通数据源，为用户提供当前道路拥堵情况，路线及时调整。  
  
 - \*\*接口类型\*\*: RESTful API, Web socket API  
 - \*\*通信协议\*\*: HTTP/HTTPS, WebSocket  
 - \*\*数据格式\*\*: JSON  
 - \*\*功能\*\*: 实时交通信息查询、路线实时更新  
  
- \*\*交通时刻表更新API\*\*: 集成了公交、地铁及其他公共交通工具的运营时刻表更新服务，确保提供最新公交、地铁线路以及时刻表信息。  
  
 - \*\*接口类型\*\*: RESTful API  
 - \*\*通信协议\*\*: HTTP/HTTPS  
 - \*\*数据格式\*\*: JSON  
 - \*\*功能\*\*: 获取公交、地铁线路以及时刻表信息，时刻表更新  
  
- \*\*兴趣点管理服务API\*\*: 与第三方地点数据库服务集成（如景区、饭店等兴趣点），为用户提供丰富的兴趣点信息和搜索功能。  
  
 - \*\*接口类型\*\*: RESTful API  
 - \*\*通信协议\*\*: HTTP/HTTPS  
 - \*\*数据格式\*\*: JSON  
 - \*\*功能\*\*: 搜索、添加、编辑兴趣点信息，获取附近兴趣点信息  
  
- \*\*用户系统及身份验证API\*\*: 用于支持注册、登录、用户信息管理等功能，可能对接至社交登录（如微信、Facebook等）或其他身份验证机制。  
  
 - \*\*接口类型\*\*: RESTful API  
 - \*\*通信协议\*\*: HTTP/HTTPS  
 - \*\*数据格式\*\*: JSON  
 - \*\*功能\*\*: 用户注册、登录、信息管理、社交登录接口  
  
#### 4.3 数据库接口需求  
  
- \*\*数据库类型\*\*: 使用现代关系型数据库（如PostgreSQL）或NoSQL数据库（如MongoDB）存储用户信息、路线偏好、兴趣点等数据。  
  
- \*\*数据交互协议\*\*: 基于SQL或NoSQL查询语言与数据库进行数据交互。  
  
 - \*\*功能\*\*: 用户信息存储与查询、对注册用户的登录验证、路线偏好数据管理、兴趣点数据同步等  
  
- \*\*界面设计\*\*: API接口的设计需考虑数据的灵活性和可扩展性，能够随着未来增加数据库表结构和字段的变化，同时实现与第三方数据库服务的无缝对接  
  
本部分详细介绍了City Mapper在生产设计环境中对硬件、软件及数据库的需求和规范。确保了无论是在软件操作界面还是在后台数据库的相互连通方面，都能够实现高效稳定的数据交互及功能执行，以为最终用户提供一站式城市出行解决方案。

# Non-Functional Requirement

### 第五章 非功能需求  
  
本章节旨在全面描述City Navigator系统的非功能需求，包括性能、可用性、可靠性、安全性、可维护性、可移植性等多个方面，确保该系统能够满足用户的需求，为用户提供稳定、安全和高效的出行服务。  
  
#### 性能指标  
  
##### 响应时间  
- \*\*日常使用情况\*\*：在非高峰时段或日常使用中，系统响应时间应在1秒到2秒内。  
- \*\*高峰时段\*\*：在用户使用高峰时段，例如早晚高峰期间，系统响应时间需保证在3秒至5秒内。这包括用户查询路线、实时交通状况更新、以及兴趣点信息加载等操作。  
   
##### 吞吐量  
- 在任意时刻内，系统需支持每分钟处理5万个请求，保证全局交互流畅，确保用户在低延迟能使用City Navigator平台以最快的速度获取所需信息。  
  
##### 并发用户数  
- 系统需支持至少50,000个并发用户同时访问，确保服务器的稳定性与响应速度不受影响。  
  
#### 可用性  
系统的可用性包括在各种情况下能够正常运行的需求。  
  
- \*\*持续可用性\*\*：系统需保证99.99%的时间段内均能正常提供服务。  
- \*\*恢复时间目标（RTO）\*\*：即恢复服务的最长时间。在遇到意外停机后，系统应能在1小时内恢复至一个可以提供完整服务的状态。  
- \*\*场景模拟\*\*：系统需在不同场景下进行测试如高峰时段，突发大量流量导致服务器不堪重载、部分节点宕机等情况，保证恢复正常完整服务的RTO时间不超过2小时。  
   
#### 可靠性  
系统具有正确的实现其指定功能的能力。  
  
- \*\*数据完整性\*\*：在任何情况下，系统需保证80万以上的用户数据完整性。对于本地缓存，需确保数据的同步更新在1秒内完成，系统在接收到更新请求后能及时生产一个同步目标数据。  
- \*\*异常处理\*\*：异常情况下软实现影响最小化，能在崩溃70%节点的情况下响应成功请求、用户可以在系统不可用时在网络受限情况下查看先前保存的路线信息。  
  
#### 安全性  
特别在确保系统安全方面，我们设定严格的标准，保护用户隐私及数据安全。  
  
- \*\*用户数据安全\*\*：确保用户敏感信息如姓名，电话号码，sos紧急联系人等从存储和传输过程中的密保护。  
- \*\*防暴力破解及敏感信息提醒\*\*：账户较差密码设定，低安全性会被分类并提醒用户可以通过个人信息中心排除自行排查，非法尝试多次登录后应在15分钟内锁定5分钟，一段时间内不响应登录请求并记录锁定和再次尝试的时间，该部分支持5次登录尝试后立即锁定10分钟，然后逐步增加锁定时间。  
- \*\*隐私保护\*\*：严格遵守相关法律法规对个人信息的保护，例如仅访问用户同意的意见收集的用户轨迹数据，数据加密保护避免人前，严格限制数据流出。  
  
#### 可维护性  
- \*\*代码质量\*\*：编写代码的参考标准，应遵循最佳实践原则，如遵守单一职责原则、开发规范和编码标准，确保代码的质量高。通过自动化构建测试的实现提高代码更改的代码质量保证。  
- \*\*技术支持响应时间\*\*：所有涉及软件产品服务中遇到的问题必须在24小时内得到响应，取消锁定次数避免较低的人为操作。  
- \*\*文档管理和维护\*\*：系统的设计文档和用户说明书内容清晰表述，条理化避免冗余性，文档更新和升级版本机制应跟上开发流程中同步更新。  
  
#### 可移植性  
- \*\*平台适应性\*\*：设计平台可以使得用户在不同设备间的平滑迁移，便于维护和升级。保证系统可运行在包括 iOS、Android、Web在内的各种主流平台间无缝使用。  
- \*\*用户数据迁移\*\*：确保用户可以在不同类型的设备上通过具体信息识别和自动账号关联操作，迁移数据时实现数据完整性。  
  
#### 可扩展性  
- \*\*处理性能的扩展性\*\*：设计系统架构以提前考虑未来的负载增加，保证平台可以承担更高的并发访问量和更苛刻的性能需求时，支持组件式无缝嵌入技术进行弹性伸缩，例如在服务器负载达到 80% 时触发自动实例扩展服务。  
- \*\*模块化设计\*\*：软硬件更新的同时保障系统具备可扩展性，例如增加服务或更改服务能带来额外功能，充分考虑未来可能对系统进行的扩展，如地铁路线全新开通后，系统将在7天内实现实时更新；如新交规的生效，系统应及时更新地图中路线。  
- \*\*系统接口的拓展性\*\*：确保系统可以通过标准的API接口联接其他第三方服务，如地图服务提供商、报告和报告等，同时可以为用户提供可自定义开发，方便快速或定制服务开发。  
  
确保系统各方面的非功能需求被完整地定义和约束，可以为 City Navigator 的软件开发过程提供一个更加系统化、明确化和高效化的指导原则。这不仅有利于确保软件产品的质量和稳定性，还能有效提升用户的整体满意度和依赖度。在这一系列非功能需求的标准约束和严格测试下，City Navigator 将成为一个易于管理和长期稳定运行的城市交通导航平台。