# Functional Requirement

- 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。  
  
-创建起始点  
 描述：用户可创建新的起始点坐标和地址信息。  
 输入：起始点的详细信息（如地址、名称）。  
 输出：数据库存储用户创建的起始信息。  
  
-管理起始点  
 描述：管理员可更新或删除系统中的起始点。  
 输入：起始点的标识符、需要更新的信息或删除的操作。  
 输出：系统中的起始点信息更新或被删除。  
  
-创建目的地  
 描述：用户可创建新的目的地坐标和地址信息。  
 输入：目的地的详细信息（如地址、名称）。  
 输出：数据库存储用户创建的目的地信息。  
  
-管理目的地  
 描述：管理员可更新或删除系统中的目的地。  
 输入：目的地的标识符、需要更新的信息或删除的操作。  
 输出：系统中的目的地信息更新或被删除。  
  
-规划路线  
 描述：用户可以根据起点和目的地信息获取由City Mapper应用推荐的路线列表。  
 输入：起点和目的地的详细信息。  
 输出：系统提供的起点到目的地的推荐路径。  
  
-选择交通方式  
 描述：用户可以选择特定的交通方式参与路线规划。  
 输入：用户的选择（如公共交通、步行、骑行等）。  
 输出：系统存储用户的选择，并作为进一步分析的基础。  
  
-距离查询  
 描述：用户可以查询起始点和目的地之间的直线或道路距离。  
 输入：起始点和目的地的详细信息。  
 输出：两地之间的距离信息，以千米或英里表示。  
  
-计算旅行时间  
 描述：用户可以计算从起始点到目的地的预计旅行时间，基于选择的交通方式。  
 输入：起始点、目的地、和交通方式的详细信息。  
 输出：预计的旅行时间。  
  
-设置用户偏好  
 描述：用户可以设置其在交通方式和路线上的偏好。  
 输入：用户偏好信息（如偏好使用公共交通而非单车）。  
 输出：存储于数据库中的用户偏好设置。  
  
-实时更新交通状况  
 描述：系统将实时更新交通状况，并反映在应用中。  
 输入：来自外部的实时交通数据源。  
 输出：数据库中最新的交通状况信息。  
  
-查询时刻表  
 描述：用户可以查询相关交通方式的时刻表信息。  
 输入：查询条件，例如特定交通方式、站点名称和日期等。  
 输出：符合查询条件的时刻表信息。  
  
-探索兴趣点  
 描述：用户可以浏览或搜索各种兴趣点，如景点、餐馆等。  
 输入：兴趣点的查询条件。  
 输出：系统提供的搜索结果，包括兴趣点的名称、地址和类别等信息。  
  
-创建和管理管理员账户  
 描述：为系统管理员创建账户并进行账户管理（更新或删除操作）。  
 输入：管理员账户的详细信息，更新或删除操作。  
 输出：系统中管理员账户信息的更新或删除。  
  
-用户偏好记录  
 描述：系统记录并持续更新用户偏好设置。  
 输入：用户的偏好选择。  
 输出：存储在数据库中的用户偏好记录。  
  
-实时交通状况记录  
 描述：系统存储和更新实时交通状况数据。  
 输入：交通状况的数据源信息。  
 输出：数据库中最新的实时交通状况记录。

# External Description

#### 4.1 用户接口（UI）  
  
本节概述软件与最终用户之间的接口。用户通过图形用户界面（UI）与软件交互。City Mapper的设计必须满足跨平台兼容性，提供一致且优化的用户体验。  
  
- 操作界面：包括但不限于创建和管理起始点及目的地、选择交通方式、查询距离及旅行时间、设置用户偏好以及探索兴趣点等功能的部分。  
- 界面响应性：UI设计需要确保快速响应用户的交互，支持多种输入方法，如触摸、点击和手势。  
- 适应性设计：确保UI在不同设备中和操作系统中的一致性显示，并且支持可定制的视图选项来优化不同用户的需求。  
- 数据展示：将实时交通状况、导航路线规划和其他重要信息以用户友好且直观的形式展现，提供信息图表、地图和其他可视化工具。  
  
#### 4.2 系统接口（API）  
  
City Mapper与外部系统（例如，地图API、交通实时数据API）交互，以提供导航和交通信息服务。这些接口是实现系统核心功能的关键。  
  
- 地图服务API：为路线规划和展示用户当前位置提供地图视图。  
 - 通信协议： RESTful API通过HTTP/HTTPS进行通信。  
 - 数据格式： JSON用于层端数据传输。  
 - 使用案例：规划最优出行路线。  
- 实时交通信息API：获取起点和目的地附近的实时交通状况，帮助用户进行路径优化。  
 - 通信协议： RESTful API通过HTTP/HTTPS进行通信。  
 - 数据格式： JSON。  
 - 使用案例：实时更新地图上交通拥堵的路段信息。  
- 数据库接口：系统使用数据库来存储和检索用户数据、偏好设置、交通时刻表等信息。  
 - 数据库类型：利用关系型数据库(SQL)和/或NoSQL数据库（例如MongoDB或Cassandra）根据具体需求进行了适当的选择。  
 - 访问协议：应用SQL语句或者采用NoSQL数据库特有的API进行数据查询、修改操作。  
 - 数据格式： SQL语句，以及NoSQL适用数据模型。  
  
#### 4.3 数据接口  
  
City Mapper不仅依赖内部数据库存储数据，还需要外部实时数据源获取动态信息，比如交通实时状况的更新，此类外部数据源头包括但不限于城市交通管理中心或者其他第三方通讯公司。  
  
- 数据交换格式：优先选择开源标准和广泛采用的数据交换格式，比如XML, JSON, CSV等进行数据交换以增强兼容性。  
- 获取数据：通过API接口（例如REST，WebSocket等）来实现实时信息更新，确保数据的实时性和准确性。  
- 数据更新频率：实时数据的更新频率依赖于数据源以及系统的特定需求。一般而言，针对实时交通数据，更新频率可达到每分钟甚至更短。  
  
#### 4.4 安全接口  
  
系统对外接口需具备高水平的安全性，确保用户信息不被泄露及数据传输的安全性。  
  
- 安全机制：借助SSL加密技术保护用户数据传输的隐私。  
- 标准遵循：严格遵守ISO/IEC 27001信息安全管理体系标准，进行风险评估来检查及减少任何潜在的安全漏洞。  
- 用户权限管理：确保只有授权人员才能访问某些系统功能，比如管理起始点或目的地、记录用户偏好等。  
  
#### 4.5 可访问性接口  
  
保证软件产品可以为所有用户提供无障碍访问，无论用户的物理、感官或认知障碍。  
  
- 文本到语音：对于视觉有障碍的用户，提供文本转语音的支持。  
- 语音控制：允许用户通过语音输入互动，便于操作软件界面，例如通过Siri, Google Assistant等。  
- 放大选项：图形和其他UI元素必须能够被放大或缩小，以满足有不同视觉需求的用户。