DirectionMQ

•••

Alvin Saisanavong, Sonny Sayaloune ,Ethan almonte, Yuvraj Gill

Vision Statement

DirectionMQ will be a comprehensive navigation tool tailored for:

- **Students**
- **Teaching Staff**
- **Visitors**

At Macquarie University Campus.

Our goal is to provide our users with the most convenient and smoothest navigation experience while on our campus.

System Architecture

Overview:

High-level diagram showing the architecture of DirectionsMQ.

Components:

Client App: Mobile application for user interaction**.**

Backend Server: Handles user requests and data management**.**

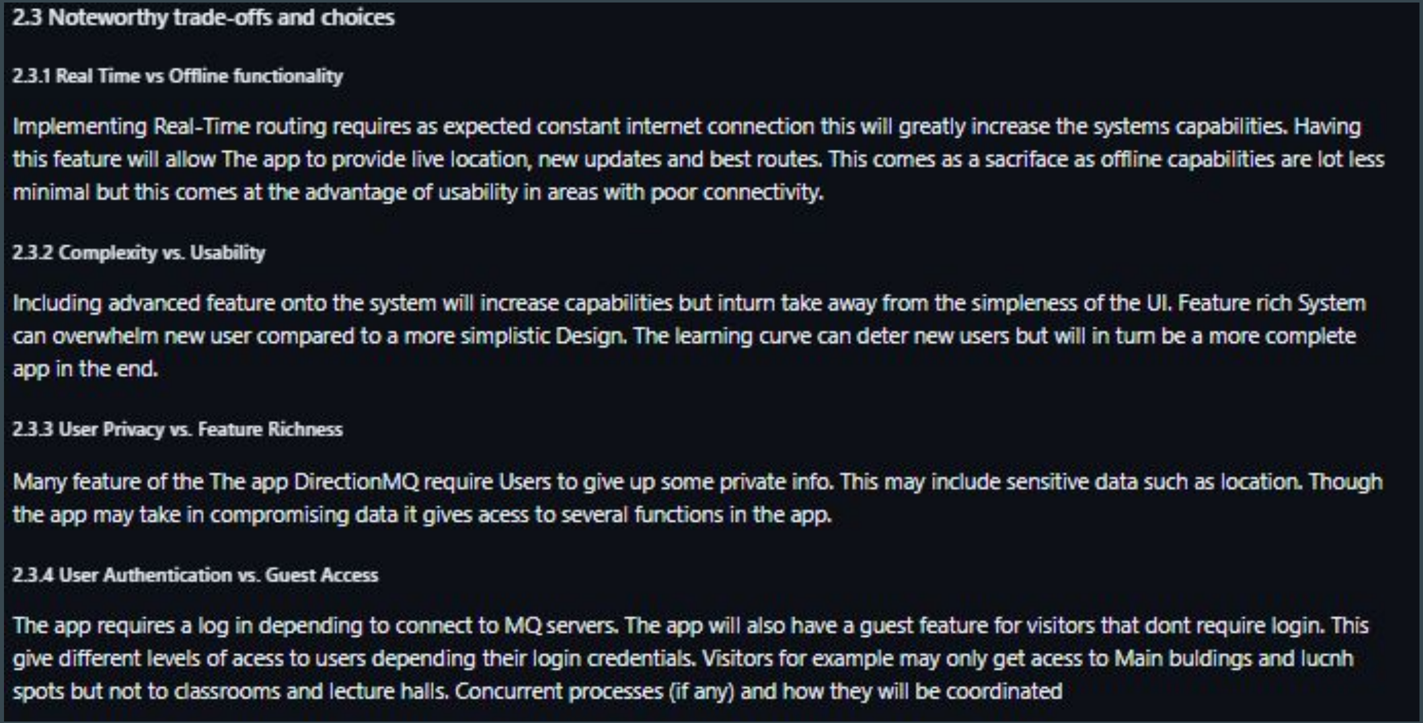
Mapping API: External service for mapping and navigation.

Database: Stores user data, waypoints, and application settings.

Flow of Information:

User actions -> Client App -> Backend Server -> Mapping API -> Response to Client App.

System Architecture - Trade off and choices



2.3 Noteworthy trade-offs and choices

2.3.1 Real Time vs Offline functionality

Implementing Real-Time routing requires as expected constant internet connection this will greatly increase the systems capabilities. Having this feature will allow The app to provide live location, new updates and best routes. This comes as a sacriface as offline capabilities are lot less minimal but this comes at the advantage of usability in areas with poor connectivity.

2.3.2 Complexity vs. Usability

Including advanced feature onto the system will increase capabilities but inturn take away from the simpleness of the UI. Feature rich System can overwhelm new user compared to a more simplistic Design. The learning curve can deter new users but will in turn be a more complete app in the end.

2.3.3 User Privacy vs. Feature Richness

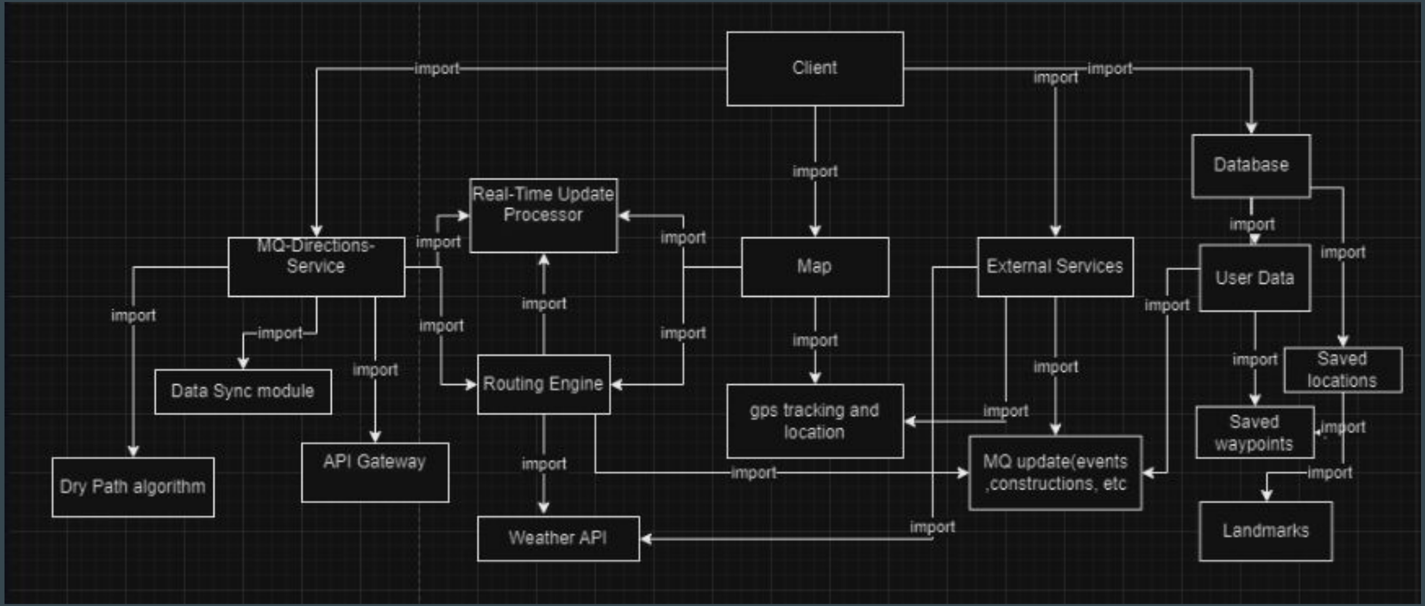
Many feature of the The app DirectionMQ require Users to give up some private info. This may include sensitive data such as location. Though the app may take in compromising data it gives acess to several functions in the app.

2.3.4 User Authentication vs. Guest Access

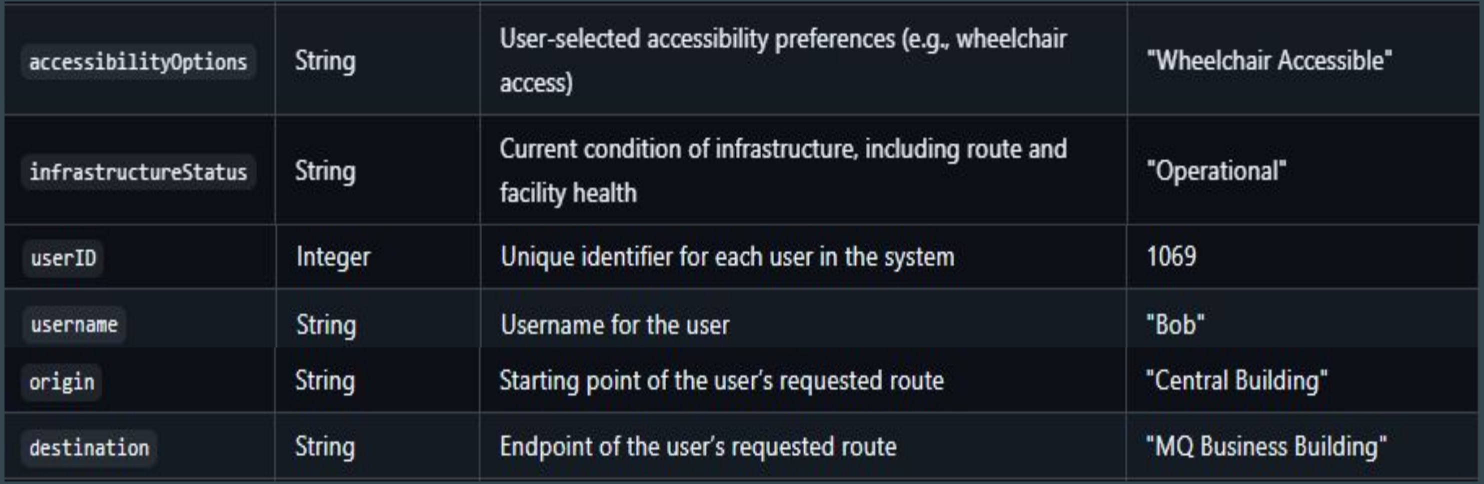
The app requires a log in depending to connect to MQ servers. The app will also have a guest feature for visitors that dont require login. This give different levels of acess to users depending their login credentials. Visitors for example may only get acess to Main buldings and lucnh spots but not to classrooms and lecture halls. Concurrent processes (if any) and how they will be coordinated

System Architecture

Package Diagram



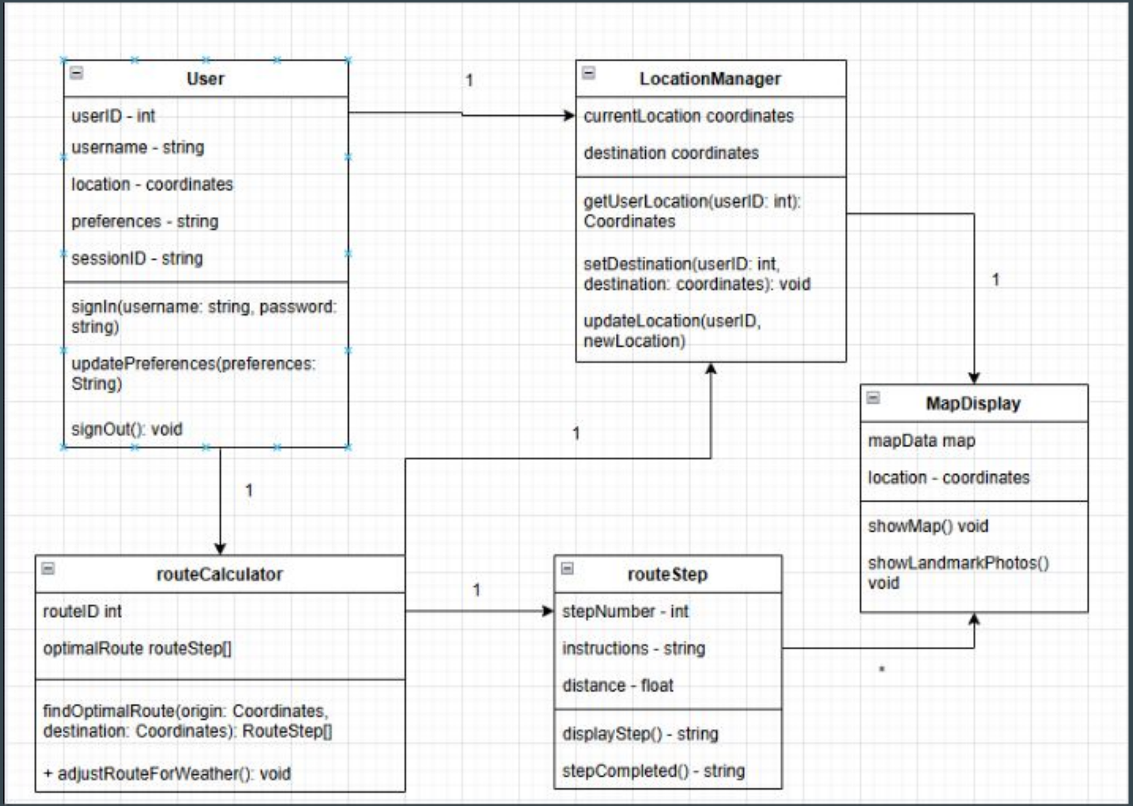
System Architecture - Data Definitions Snippet



`accessibilityOptions`	String	User-selected accessibility preferences (e.g., wheelchair access)	"Wheelchair Accessible"
`infrastructureStatus`	String	Current condition of infrastructure, including route and facility health	"Operational"
`userID`	Integer	Unique identifier for each user in the system	1069
`username`	String	Username for the user	"Bob"
`origin`	String	Starting point of the user's requested route	"Central Building"
`destination`	String	Endpoint of the user's requested route	"MQ Business Building"

Summary of Relationships

- **User to LocationManager**: User's location management.
- **User to routeCalculator**: Access to route calculation.
- **LocationManager to MapDisplay**: Updates map display based on user’s position.
- **LocationManager to routeCalculator**: Provides location for route planning.
- **routeCalculator to routeStep**: Defines route in sequential steps.



User

userID - int

username - string

location - coordinates

preferences - string

sessionID - string

signIn(username: string, password: string)

updatePreferences(preferences: String)

signOut(): void

LocationManager

currentLocation coordinates

destination coordinates

getUserLocation(userID: int): Coordinates

setDestination(userID: int, destination: coordinates): void

updateLocation(userID, newLocation)

MapDisplay

mapData map

location - coordinates

showMap() void

showLandmarkPhotos() void

routeCalculator

routeID int

optimalRoute routeStep[]

findOptimalRoute(origin: Coordinates, destination: Coordinates): RouteStep[]

+ adjustRouteForWeather(): void

routeStep

stepNumber - int

instructions - string

distance - float

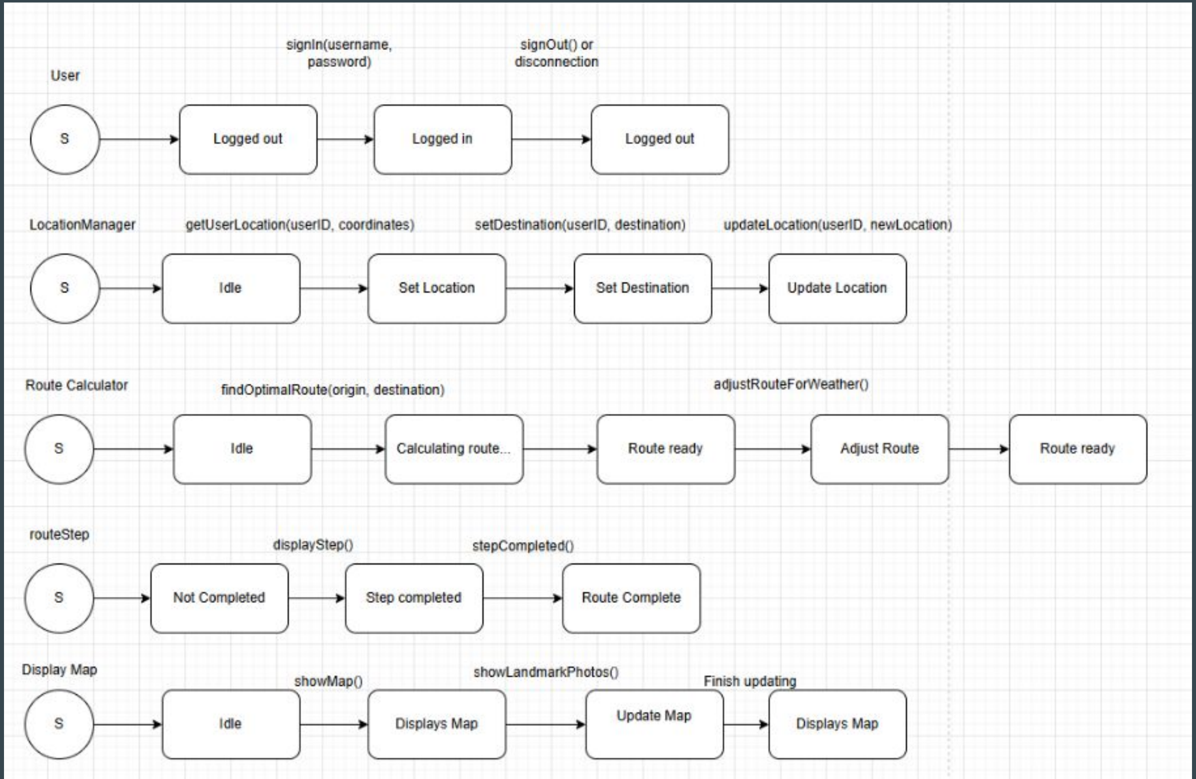
displayStep() - string

stepCompleted() - string

Analysis and Design - Class Diagram

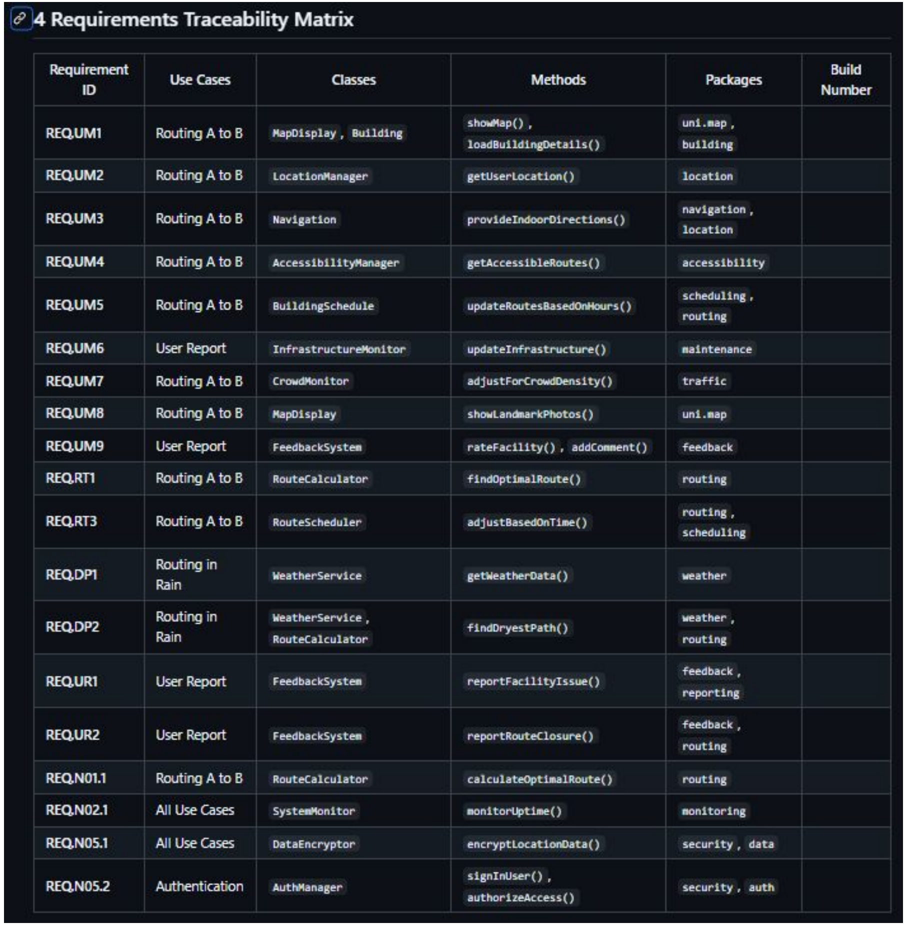
Summary of Component Interactions

- **User** initiates login and interacts with **LocationManager** for setting locations.
- **LocationManager** updates **Route Calculator** and **Map Display** as the user progresses.
- **Route Calculator** calculates routes and interacts with **routeStep** for step tracking.
- **Display Map** provides a real-time view, updating based on user movement.
- Together, these state transitions create a smooth and dynamic navigation experience.



Analysis and Design - State Diagrams

Traceability

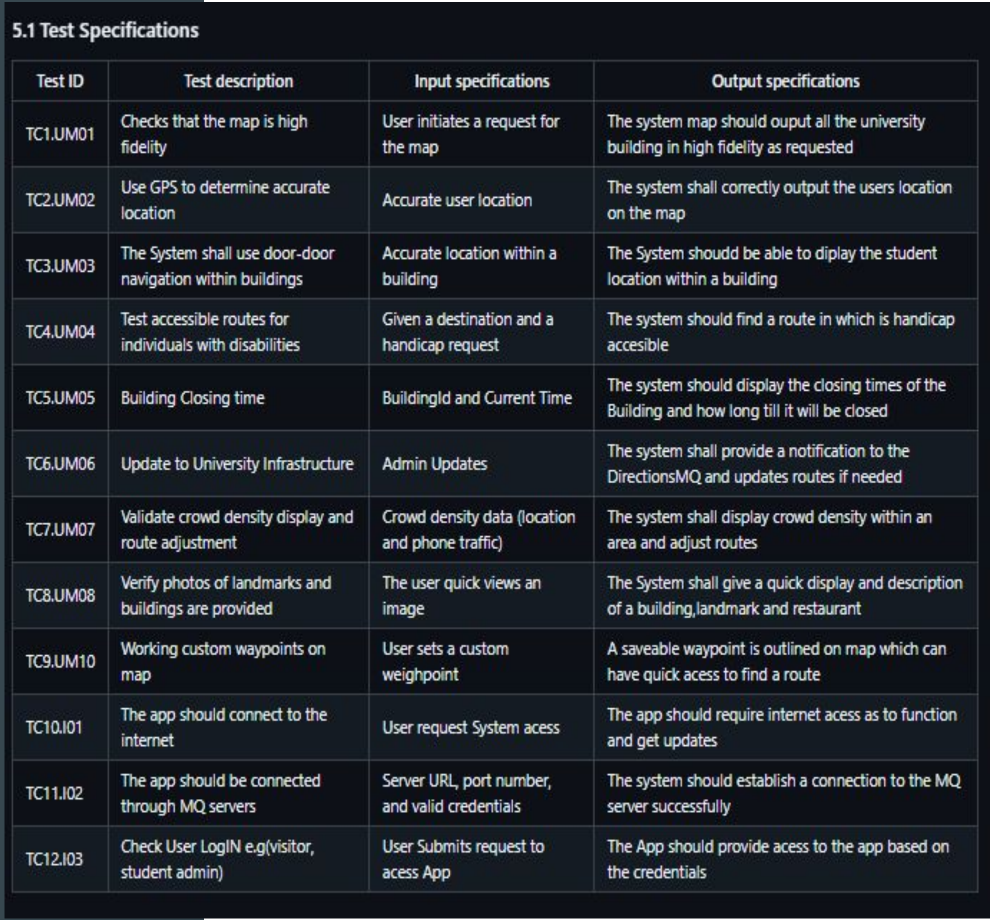


4 Requirements Traceability Matrix

Requirement ID	Use Cases	Classes	Methods	Packages	Build Number
REQ.UM1	Routing A to B	`MapDisplay`, `Building`	`showMap()`, `loadBuildingDetails()`	`uni.map`, `building`	
REQ.UM2	Routing A to B	`LocationManager`	`getUserLocation()`	`location`	
REQ.UM3	Routing A to B	`Navigation`	`provideIndoorDirections()`	`navigation`, `location`	
REQ.UM4	Routing A to B	`AccessibilityManager`	`getAccessibleRoutes()`	`accessibility`	
REQ.UM5	Routing A to B	`BuildingSchedule`	`updateRoutesBasedOnHours()`	`scheduling`, `routing`	
REQ.UM6	User Report	`InfrastructureMonitor`	`updateInfrastructure()`	`maintenance`	
REQ.UM7	Routing A to B	`CrowdMonitor`	`adjustForCrowdDensity()`	`traffic`	
REQ.UM8	Routing A to B	`MapDisplay`	`showLandmarkPhotos()`	`uni.map`	
REQ.UM9	User Report	`FeedbackSystem`	`rateFacility()`, `addComment()`	`feedback`	
REQ.RT1	Routing A to B	`RouteCalculator`	`findOptimalRoute()`	`routing`	
REQ.RT3	Routing A to B	`RouteScheduler`	`adjustBasedOnTime()`	`routing`, `scheduling`	
REQ.DP1	Routing in Rain	`WeatherService`	`getWeatherData()`	`weather`	
REQ.DP2	Routing in Rain	`WeatherService`, `RouteCalculator`	`findDryestPath()`	`weather`, `routing`	
REQ.UR1	User Report	`FeedbackSystem`	`reportFacilityIssue()`	`feedback`, `reporting`	
REQ.UR2	User Report	`FeedbackSystem`	`reportRouteClosure()`	`feedback`, `routing`	
REQ.N01.1	Routing A to B	`RouteCalculator`	`calculateOptimalRoute()`	`routing`	
REQ.N02.1	All Use Cases	`SystemMonitor`	`monitorUptime()`	`monitoring`	
REQ.N05.1	All Use Cases	`DataEncryptor`	`encryptLocationData()`	`security`, `data`	
REQ.N05.2	Authentication	`AuthManager`	`signInUser()`, `authorizeAccess()`	`security`, `auth`	

Test

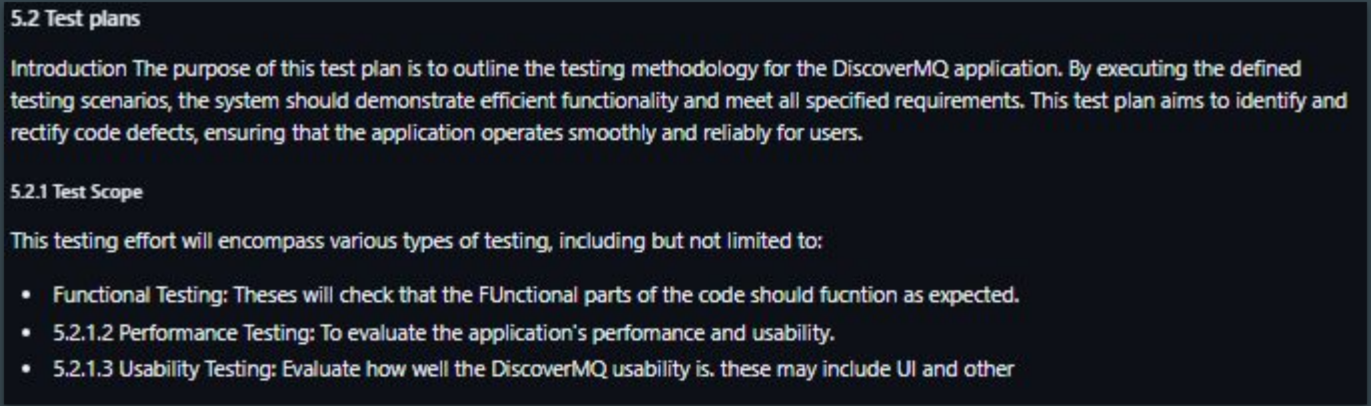
Test Specifications



5.1 Test Specifications

Test ID	Test description	Input specifications	Output specifications
TC1.UM01	Checks that the map is high fidelity	User initiates a request for the map	The system map should ouput all the university building in high fidelity as requested
TC2.UM02	Use GPS to determine accurate location	Accurate user location	The system shall correctly output the users location on the map
TC3.UM03	The System shall use door-door navigation within buildings	Accurate location within a building	The System shoudd be able to diplay the student location within a building
TC4.UM04	Test accessible routes for individuals with disabilities	Given a destination and a handicap request	The system should find a route in which is handicap accesible
TC5.UM05	Building Closing time	BuildingId and Current Time	The system should display the closing times of the Building and how long till it will be closed
TC6.UM06	Update to University Infrastructure	Admin Updates	The system shall provide a notification to the DirectionsMQ and updates routes if needed
TC7.UM07	Validate crowd density display and route adjustment	Crowd density data (location and phone traffic)	The system shall display crowd density within an area and adjust routes
TC8.UM08	Verify photos of landmarks and buildings are provided	The user quick views an image	The System shall give a quick display and description of a building,landmark and restaurant
TC9.UM10	Working custom waypoints on map	User sets a custom weighpoint	A saveable waypoint is outlined on map which can have quick acess to find a route
TC10.I01	The app should connect to the internet	User request System acess	The app should require internet acess as to function and get updates
TC11.I02	The app should be connected through MQ servers	Server URL, port number, and valid credentials	The system should establish a connection to the MQ server successfully
TC12.I03	Check User LogIN e.g(visitor, student admin)	User Submits request to acess App	The App should provide acess to the app based on the credentials

Test plan



5.2 Test plans

Introduction The purpose of this test plan is to outline the testing methodology for the DiscoverMQ application. By executing the defined testing scenarios, the system should demonstrate efficient functionality and meet all specified requirements. This test plan aims to identify and rectify code defects, ensuring that the application operates smoothly and reliably for users.

5.2.1 Test Scope

This testing effort will encompass various types of testing, including but not limited to:

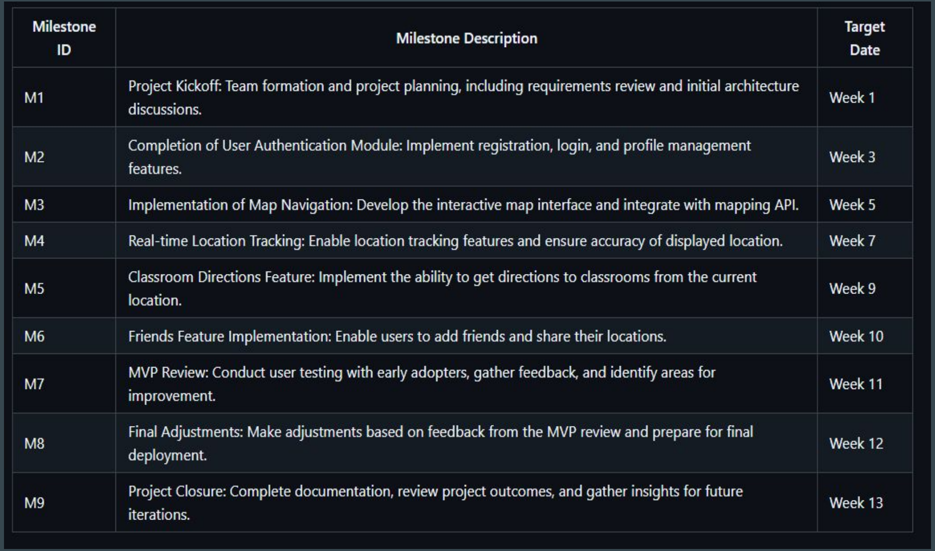
- Functional Testing: Theses will check that the FUnctional parts of the code should fucntion as expected.
- 5.2.1.2 Performance Testing: To evaluate the application's perfomance and usability.
- 5.2.1.3 Usability Testing: Evaluate how well the DiscoverMQ usability is. these may include UI and other

Project Management

Minimal Viable Product (MVP)

- User Authentication: Secure account creation and login (R1)
- Real-time Location Tracking: Display user's current location (R2)
- Turn-by-Turn Navigation: Guide users to selected destinations (R3)
- Waypoint Management: Save and manage frequently visited locations (R4)
- Location Sharing: Share location with friends (R5)

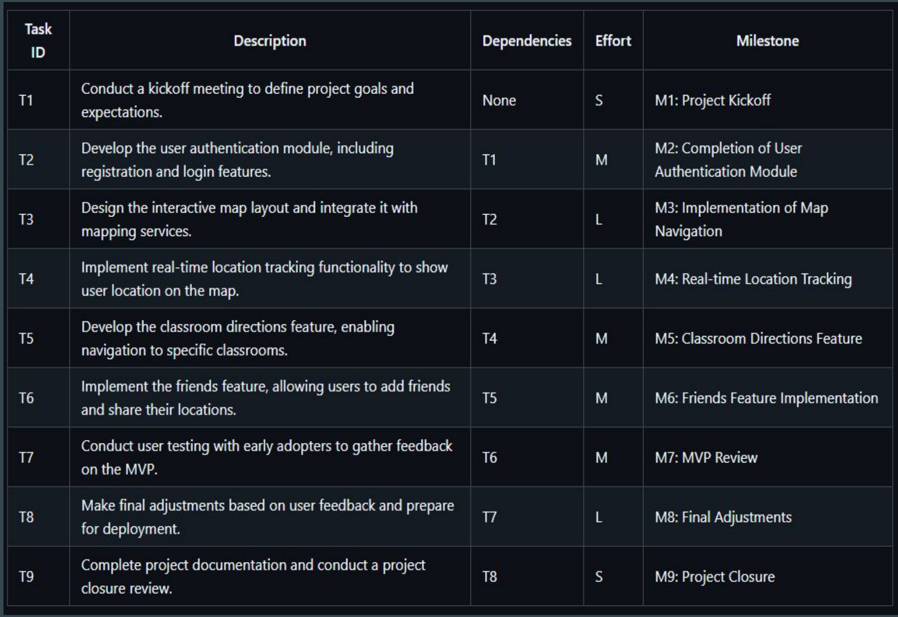
Milestones



Milestone ID	Milestone Description	Target Date
M1	Project Kickoff: Team formation and project planning, including requirements review and initial architecture discussions.	Week 1
M2	Completion of User Authentication Module: Implement registration, login, and profile management features.	Week 3
M3	Implementation of Map Navigation: Develop the interactive map interface and integrate with mapping API.	Week 5
M4	Real-time Location Tracking: Enable location tracking features and ensure accuracy of displayed location.	Week 7
M5	Classroom Directions Feature: Implement the ability to get directions to classrooms from the current location.	Week 9
M6	Friends Feature Implementation: Enable users to add friends and share their locations.	Week 10
M7	MVP Review: Conduct user testing with early adopters, gather feedback, and identify areas for improvement.	Week 11
M8	Final Adjustments: Make adjustments based on feedback from the MVP review and prepare for final deployment.	Week 12
M9	Project Closure: Complete documentation, review project outcomes, and gather insights for future iterations.	Week 13

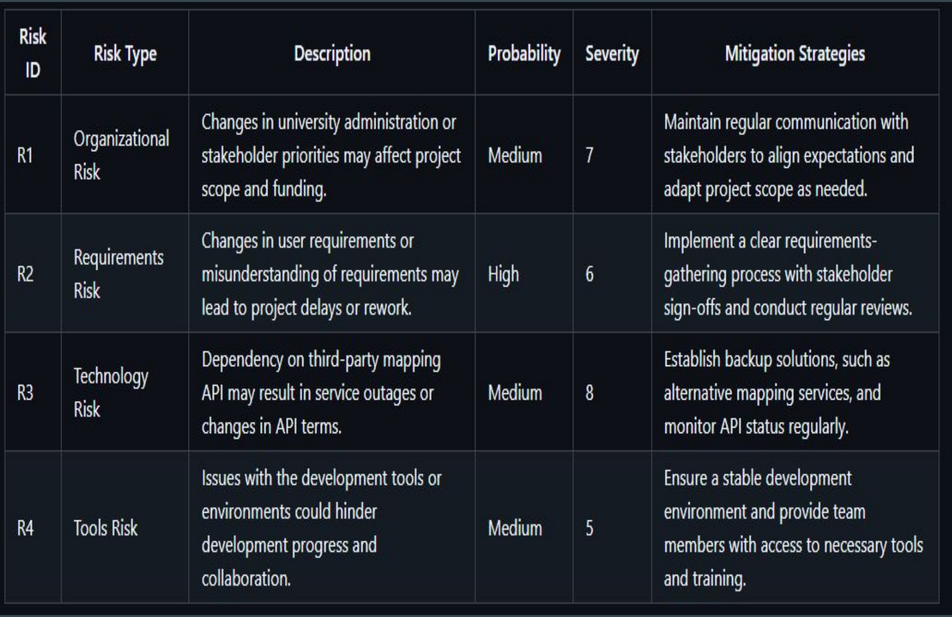
Project Management

Task



Task ID	Description	Dependencies	Effort	Milestone
T1	Conduct a kickoff meeting to define project goals and expectations.	None	S	M1: Project Kickoff
T2	Develop the user authentication module, including registration and login features.	T1	M	M2: Completion of User Authentication Module
T3	Design the interactive map layout and integrate it with mapping services.	T2	L	M3: Implementation of Map Navigation
T4	Implement real-time location tracking functionality to show user location on the map.	T3	L	M4: Real-time Location Tracking
T5	Develop the classroom directions feature, enabling navigation to specific classrooms.	T4	M	M5: Classroom Directions Feature
T6	Implement the friends feature, allowing users to add friends and share their locations.	T5	M	M6: Friends Feature Implementation
T7	Conduct user testing with early adopters to gather feedback on the MVP.	T6	M	M7: MVP Review
T8	Make final adjustments based on user feedback and prepare for deployment.	T7	L	M8: Final Adjustments
T9	Complete project documentation and conduct a project closure review.	T8	S	M9: Project Closure

Risk



Risk ID	Risk Type	Description	Probability	Severity	Mitigation Strategies
R1	Organizational Risk	Changes in university administration or stakeholder priorities may affect project scope and funding.	Medium	7	Maintain regular communication with stakeholders to align expectations and adapt project scope as needed.
R2	Requirements Risk	Changes in user requirements or misunderstanding of requirements may lead to project delays or rework.	High	6	Implement a clear requirements-gathering process with stakeholder sign-offs and conduct regular reviews.
R3	Technology Risk	Dependency on third-party mapping API may result in service outages or changes in API terms.	Medium	8	Establish backup solutions, such as alternative mapping services, and monitor API status regularly.
R4	Tools Risk	Issues with the development tools or environments could hinder development progress and collaboration.	Medium	5	Ensure a stable development environment and provide team members with access to necessary tools and training.

Summary

The DirectionsMQ project focuses on developing a navigation and location-sharing app for the Macquarie University community. The MVP will provide essential features like user authentication, real-time location tracking, and turn-by-turn navigation to meet the basic needs of students and staff.

Moving forward, we will prioritize refining the app based on user feedback from the MVP review, ensuring it evolves to enhance the user experience. Future enhancements may include personalized route recommendations and integration with university events.

Ultimately, DirectionsMQ aims to streamline navigation on campus and foster a stronger sense of community among users, positioning itself as a vital tool within the university environment.