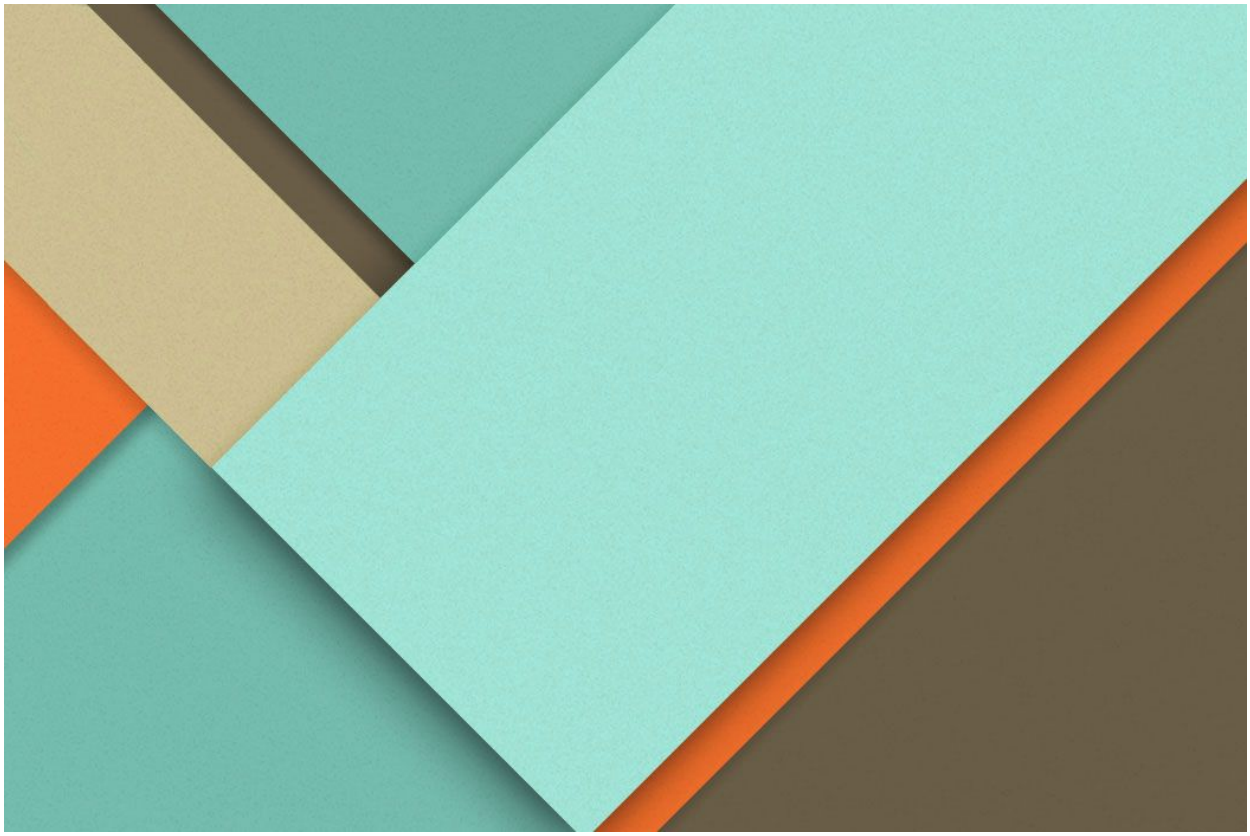


Student Travel App

Software Requirement Specification

07.03.2018

Group 9

Ankush Phulia (2014CS50279)

Deepak Saini (2013CS50281)

Shashank Yadav (2013CS50799)

Vikas Chouhan (2014CS10264)

Table of Contents

1 Introduction

Majority of the students in IIT live quite far away from campus. More often than not, when they plan a trip to their home, they do it alone, and might end up meeting someone coincidentally on their way. Further when students plan trips, it is often difficult to do due to all the work involved - research on destination, cost and reaching consensus. In this document we intend to tackle these problems, and propose a seamless solution.

1.1 Purpose

The purpose of this document is to detail the requirements of “Student Travel App”, a seamless framework that allows users to plan a journey or trip and coordinate, based on the time of journey and destination. It will illustrate the purpose and complete declaration for the development of system. It will also explain system constraints, interface and interactions with other external applications.

1.2 Scope

“Student Travel App” is intended to be a free-to-use application that will be -

1. An easy to use framework, that allows planning journeys to and from a user’s hometown(final destination), from and to user’s hostel(initial destination)
2. A planner for organising a collective trip to a destination - consensus on plan, travel to and stay at the destination

Main aim is to coordinate the journeys of people based on the time of journey and destination so that students can travel together to make for a safer and more enjoyable journey. There is a host of features planned to streamline and improve user experience,, including tracking payments made, real-time notifications, rating/review system for destinations and journeys, based on aggregating user responses.

1.3 Overview

The remainder of the document is organised into two sections - the Overall Description and the Requirements Specification. The first one provides an overview of the system functionality, system interactions and mentions the system constraints and assumptions. The second section describes the requirements specification in detailed terms and the features as stimulus-response pairs for each of these.

1.4 Definitions, Acronyms and Abbreviations

Term	Definition
User/Normal User	Someone who interacts with the application
Admin/Administrator	System administrator who is given specific permission for managing and controlling the system
Destination	The end point of any travel undertaken
Journey	A one-way travel from hostel/hometown to destination
Trip	A collective travel and stay to a particular destination
GPS	Global Positioning System, provides locations for Map/Tracking
ID	Unique identification of the requirement
DESC	Description of the requirement
RAT	Rationale behind the requirement
DEP	Dependencies, other requirements on which this requirement depends
GIST	A summary of the non-functional requirements
SCALE	Scale at which the non-functional requirement is measured
METER	Method of measuring the non-functional requirement
MUST	Minimum level of acceptable performance

1.5 References

1. IEEE Software Engineering Standards Committee, “IEEE Std 830-1998, IEEE Recommended Practice for Software Requirements Specifications”, 1998

2 Overall Description

2.1 Product Perspective

The system proposed aims to be a self-contained and independent framework(aside from using various external APIs). IRCTC's e-ticketing system provides rail travel plans. Similarly private apps like yatra, makemytrip, etc. also plan trips, but usually only for the user.

2.1.1 System Interfaces

The application will need to communicate to external map APIs, which may communicate with a physical GPS device to find/track location of the user. The map service will feed data to the app, which will use it to track and inform user about the status of the journey.

2.1.2 User Interfaces

The Front-end for the web app will be developed in HTML, CSS and Javascript(or equivalent scripting language). Potential mobile apps, would need an android smartphone.

2.1.3 Hardware Interfaces

An operating system like Linux, Android, etc capable of running modern web browsers. For running the mobile app, an android smartphone should be sufficient.

2.1.4 Software Interfaces

A browser that supports HTML, CSS & Javascript for the web app is all that is required for a normal user. Most services used will be on cloud/hosted online. External APIs like Google Maps will be required - but not needed separately from the app

2.1.5 Communication

The app will primarily be web-based and will server-client communication will be purely over the internet, subject to protocols like TCP/IP, local network protocols, etc.

2.1.6 Memory

An online database to store user and application data. It will based on a cloud platform, and likely NoSQL. The app will be served data from this database, and this will be modified.

2.1.7 Operations

The app can be run in two modes - user and admin. The user mode will expose functions for normal users, and for that particular user only. The admin mode will expose additional functionality for overall management of user and other data.

2.2 Product Functions

The primary functions of the framework are - journey planning and trip organisations, which broadly utilise/offer the following functionality

- Journey planning/scheduling
    - Routing of journey - From start to destination, mode of transport
    - Determination of Journey Overlap - between two journeys, done by system
    - Creation of a combined schedule - when parties agree to travel together
- Trip organisation
    - Trip scheduling - the destination and time
    - Consensus accumulation - from various parties for planned trip
    - Trip Invitation - new users added to planned trip
- Inter-User communication
    - Notification - when another user joins user's trip or can share journey
    - Frequent companions - quick list of most common companions
- Post-journey/Trip
    - Completion of travel - marking journey as done
    - Rating - of journey, destination and travel companion(s)

2.3 User Characteristics

There will be two kinds of users interacting with the system - normal users and admins.

The normal users will primarily be college-going students, with the initial target audience being IIT's student body. This is mainly to allow collective trips and journeys.

From a simple trip planning perspective, anyone can register and use the app. However, the collective travel options may be limited to someone from outside.

The admins will work from behind-the-scenes, and will manage the overall system to ensure consistency of information regarding users, destinations and journeys

2.4 Constraints

The App requires certain permissions, and is constrained by them

- Continuous access to the internet - fetching data from online database
- Platform limitations - for maximum compatibility with web browsers, mobile, etc.
- Database - Scalability with respect to planning/tracking multiple ongoing trip/journeys and stability
- User privacy requirements - for their profiles and journeys
- Use of external APIs (Google Maps, etc) - limit user data provided to them
- Continuous External API availability - for E-ticketing, tracking, etc.

2.5 Assumptions and Dependencies

Given the intent is for a small WebApp/Mobile App, it is assumed that

- The web app will be the primary focus, akin to IRCTC's system
- Users will have stable access to the internet when using the app. While the data will be persistent, without stable access, it would be impossible to view
- Users will have a modern browser, with Javascript enabled, for web app
- Users will have a modern smartphone device, in case of mobile app
- Data will be secure, on-cloud and will be stored reliably. Data backup and persistence will be handled by the database

3 Specific Requirements

This section contains all of the functional and quality requirements of the system. It gives a detailed description of the system and all its features.

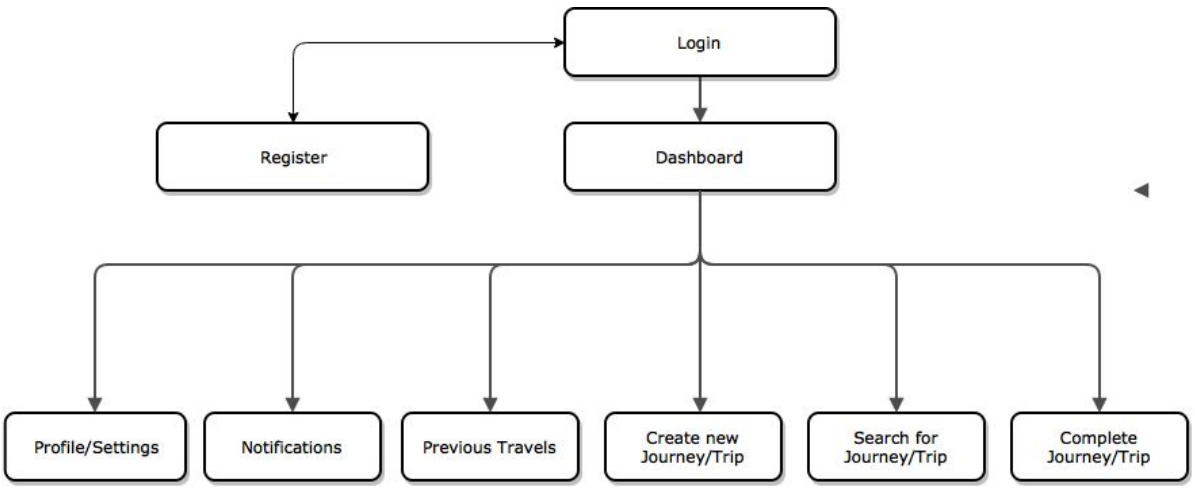
3.1 External interface requirements

This section provides a detailed description of all inputs into and outputs from the system. It also gives a description of the hardware, software and communication interfaces and provides basic prototypes of the user interface.

3.1.1 User interfaces

The currently planned user interface will consist of multiple screens

- Login - To allow users/admins to log into the system
- Register - To allow registry of new users
- Dashboard - Shows current travel progress(if any), a homepage
- Profile - Details of the logged-in user, and other settings
- Previous trips/Journeys - A record of previously completed/cancelled travels
- Notifications- A page for notifications
- New Journey/Trip creator - Create a new journey/trip object with some preliminary details, options on allowing other to join, etc.
- Search for existing Journeys/Trips - Join trip/journeys that were created by other users before and are accepting new companions
- Complete journey/trip - mark travel as finished and rate various aspects



User Interface Tree

3.1.2 Hardware interfaces

In case of web app, the app is displayed on a browser, which is to handle the rendering on the screen. Since neither the web nor the mobile app have any designated hardware, it does not have any direct hardware interfaces. The physical GPS is managed by the GPS application in the mobile phone and the hardware connection to the database server is managed by the underlying operating system on the phone.

3.1.3 Software interfaces

We have interface to two set of software API's:

- Google maps API : It is used for the display of maps in the application and for tracking the progress of trip, etc.
- eRail API : Used to get the details of train - timings, cost, availability, etc.

3.1.4 Communications interfaces

The communication between the different parts of the system is important since they depend on each other. However, in what way the communication is achieved is not important for the system and is therefore handled by the underlying internet connection protocol or operating system

3.2 Functional Requirements

This section includes the requirements that specify all the fundamental actions of the software system. We organise the functional requirements into three sub groups:

Please observe the distinction between a journey and a trip. We make this distinctions to provide an easy to use and understand view to the end user.

The requirements may be briefly classified into four kinds -

- User registration and logistics : These requirements pertain to the installation of the application, its update(when new versions are released), making of user profile and logging in.
- Journey Planning : A journey is meant for travel to/from hometown of the user.
- Trip Planning : A trip is meant for travel to a tourist spot. It may include the travel details, the stay details, the detailed organization of the trip.
- Additional functionality : These are services which may be used in both the journey/trip planning.

3.2.1 User Registration and Logistics

3.2.1.1 Functional requirement 1.1

ID: FR_1_1

TITLE: Download mobile application

DESC: A user should be able to download the mobile application through an application store on the mobile phone. The application should be free to download.

DEP: None

3.2.1.2 Functional requirement 1.2

ID: FR_1_2

TITLE: Download and notify users of new releases

DESC: When a new/updated version or release of the software is released, the user should check for these manually. The download of the new release should be done through the mobile phone in the same way as downloading the mobile application.

DEP: FR_1_1

3.2.1.3 Functional requirement 1.3

ID: FR_1_3

TITLE: User registration on the application

DESC: Given that a user has downloaded the mobile application, then the user should be able to register through the mobile application. The user must provide username, password and email address. The user can choose to provide phone no., facebook id, etc.

DEP: FR_1_1

3.2.1.4 Functional requirement 1.4

ID: FR_1_4

TITLE: User log-in

DESC: Given that a user has registered, then the user should be able to log in to the mobile application. The login information will be stored on the phone and in the future the user should be logged in automatically.

DEP: FR_1_1, FR_1_3

3.2.2 Journey Planning

3.2.2.1 Functional requirement 2.1

ID: FR_2_2

TITLE: Posting a journey

DESC: A User can post a "journey"(an entity of our system) on the app. A journey will have checkpoints in it. The user will have to specify the checkpoints. For example, when travelling from delhi to sikar, the checkpoints will be IIT hostel, NDLS, Jaipur railway station, Sikar. The journey will include the following information:

- The tentative date/time of journey
- The destination
- The preferred means of transport between different checkpoints.
- Preferred number of people you want to travel with

DEP: FR_1_4

3.2.2.2 Functional requirement 2.2

ID: FR_2_2

TITLE: Searching for journey

DESC: The user can search for a journey, again by providing the details as mentioned in FR_2_2. Matching of the journeys will be done. The matching journeys will be shown to the user in a list format in the app with the percentage match.

DEP: FR_1_4

3.2.2.3 Functional requirement 2.3

ID: FR_2_3

TITLE: Setting notification for a journey

DESC: The user can choose to provide details for the journey and then subscribe to notifications if someone posts an overlapping journey by specifying the amount of overlap. If some other user posts a journey with overlap greater than the specified, the user will get notification in the app. The overlap will be decided based on the parameters of the journey. More detailed information about how to find the overlap can been found in Appendix 1.

DEP: FR_1_4

3.2.2.4 Functional requirement 2.4

ID: FR_2_4

TITLE: Contacting the journey poster

DESC: The user can call OR contact through facebook, email the journey poster.

DEP: FR_2_3

3.2.2.5 Functional requirement 2.5

ID: FR_2_5

TITLE: Adding a user to the journey

DESC: The poster of the journey can add some other user to the journey. The users included in an open journey will be shown to the other users who search for a journey.

DEP: FR_2_2

3.2.2.6 Functional requirement 2.6

ID: FR_2_6

TITLE: Marking the journey complete

DESC: If the number of users included in the journey has reached the optimal as specified by the poster, the journey will be automatically closed.

The user can choose to close the journey manually also. A closed journey will not be shown to other users.

DEP: FR_2_2

3.2.2.7 Functional requirement 2.7

ID: FR_2_7

TITLE: Withdraw from the journey

DESC: The users added by the journey poster can decide to withdraw from the journey. A notification of such withdrawal will be sent to all the users included in the journey.

The number of withdrawals will be shown in the profile of a user.

DEP: FR_2_2, FR_2_5

3.2.3 Trip Planning

Several of the functional requirements for trip planning will overlap(with minor technical differences) with those of journey planning.

3.2.3.1 Functional requirement 3.1

ID: FR_3_1

TITLE: Posting a trip intent

DESC: A User should be able to post the intent of a trip on the app. This will include the following information:

- Preferred destination for the trip
- Preferred date/time of the trip
- Budget range estimate - transportation/food, etc

DEP: FR_1_4

3.2.3.2 Functional requirement 3.2

ID: FR_3_2

TITLE: Notification for similar trip intents

DESC: A User will get notification if some other user posts a trip intent which have overlap with the intent posted by the user. In the notification, the user should be shown

the overlap in an intuitive manner.

DEP: FR_3_1

3.2.3.3 Functional requirement 3.3

ID: FR_3_3

TITLE: Formation of trip planning group

DESC: Based on the trip intents posted by the users, trip planning group will be formed. The user can decide to form a group automatically where the other users with overlapping trip intents will be added or can manually add other users also.

The contact details of every member of the trip planning group will be shared will every other member of the group.

DEP: FR_3_1

3.2.3.4 Functional requirement 3.4

ID: FR_3_4

TITLE: Election of coordinator

DESC: The members of the group will elect the coordinator in a secret polling, with coin-toss tie-break. The coordinator will be the contact for the trip outside of the app.

DEP: FR_3_3

3.2.3.5 Functional requirement 3.5

ID: FR_3_5

TITLE: Posting of trip

DESC: Note the difference between a "trip intent" and a "trip". After the finalizing in the trip planning group, the coordinator may decide to post the trip on the app if they have room for more people to join into the trip. The trip will include details like the location of the trip, exact timeline of the trip - start/end/travel times, etc.

DEP: FR_3_3, FR_3_4

3.2.3.6 Functional requirement 3.6

ID: FR_3_6

TITLE: Searching for trips

DESC: All the trips posted will be publicly available to all the users, as long as the original poster specifies that they are willing to travel with companions. Further a user can search for trips based on filters, like the location, budget etc.

DEP: FR_1_4

3.2.3.7 Functional requirement 3.7

ID: FR_3_7

TITLE: Request for joining a trip

DESC: A user make a request to join a trip. The user can't negotiate the details of the trip and will have to accept the trip as is while making the request. The details of the request maker will be shared with the trip coordinator. These will include the past withdrawals.

DEP: FR_1_4

3.2.3.8 Functional requirement 3.8

ID: FR_3_8

TITLE: Approval/rejection of the request by the trip coordinator

DESC: The request will appear as a notification to the trip coordinator. The coordinator will be provided with the details of the requestor. He can make a decision to accept/reject the request after contact the person.

DEP: FR_3_7

3.2.3.9 Functional requirement 3.9

ID: FR_3_9

TITLE: Marking the trip complete

DESC: After completion of the trip, the coordinator will be required to mark the trip as completed so that it is removed from the search results to other users.

DEP: FR_3_5

3.2.3.10 Functional requirement 3.10

ID: FR_3_10

TITLE: Withdraw from the trip

DESC: Between joining the trip and completion of the trip, a trip member can decide to withdraw from the trip. A notification of such withdrawal will be sent to all the trip members. The number of withdrawals will be shown in the profile of a user.

DEP: FR_3_5

3.2.3.11 Functional requirement 3.11

ID: FR_3_11

TITLE: Providing Rating/Reviews about the trip

DESC: After completion of the trip, the coordinator can get "coordination points", after filling a small survey where he/she will be required to rate and provide comments about specific aspects of the trip. This is necessary to get data for analysis.

DEP: FR_3_9

3.2.3.12 Functional requirement 3.12

ID: FR_3_12

TITLE: See the Rating/Reviews about trip destinations, hotels/lodges

DESC: The ratings/reviews provided by the trip coordinator will be analysed and compiled so that a user can search about entities and can see the rating/reviews about it.

The destination reviews will be shown on a map.

DEP: FR_1_4

3.2.4 Additional Functionality

3.2.4.1 Functional requirement 4.1

ID: FR_4_1

TITLE: Coordination of Payment

DESC: A user can add bills involving multiple other users and split the amount payable among the users. The user will be required to

- Specify the amount and the items they paid for
- Upload the photo of the bill if available
- Tag the transaction with the journey/trip name.

The users can see their outstanding total dues.

DEP: FR_1_4

3.3 Performance requirements

The requirements in this section provide a detailed specification of the user interaction with the software and measurements placed on the system performance.

3.3.1 Usage of the search feature

ID: QR1

TITLE: Usage of the search feature

DESC: The different search options should be evident, simple and easy to understand.

RAT: In order to for a user to perform a search easily.

DEP: None

3.3.2 Usage of the results in the list view

ID: QR2

TITLE: Usage of the results in the list view

DESC: The results displayed in the list view should be user friendly and easy to understand. Selecting an element in the result list should only take one click.

RAT: In order to for a user to use the list view easily.

DEP: None

3.3.3 Usage of the result in the map view

ID: QR3

TITLE: Usage of the result in the map view

DESC: The results displayed in the map view should be user friendly and easy to understand. Selecting a pin on the map should only take one click.

RAT: In order to for a user to use the map view easily.

DEP: None

3.3.4 Response time

ID: QR4

GIST: The fastness of the search

SCALE: The response time of a search

METER: Measurements obtained from a sample of searches during testing.

MUST: No more than 2 seconds 100% of the time(standard in most search based apps).

3.3.5 System dependability

ID: QR5

GIST: The fault tolerance of the system.

SCALE: If the system loses the connection to the Internet or to the GPS device or the system gets some strange input, the user should be informed.

METER: Measurements obtained from some hours of usage during testing.

MUST: 100% of the time.

3.4 Design constraints

3.4.1 Hard Drive Space

ID: QR6

GIST: The amount of hard disk utilised by the mobile application.

SCALE: MB.

METER: Observations done from the performance log during testing

MUST: No more than 100 MB.

3.4.2 Application memory usage

ID: QR7

GIST: The amount of Operate System memory occupied by the application.

SCALE: MB.

METER: Observations done from the performance log during testing

MUST: No more than 20 MB.

Appendix 1 - Overlap between Journeys

A journey will be provided with checkpoints, i.e, important places where a change of transport is made. Further we have the prefered means of transport between each pair of checkpoints.

The estimated time of arrival at the different checkpoints can be made. In case of bus, doing this is easy assuming that there is not much waiting time for the bus. To estimate the arrival time at a checkpoint which via train, we will need to have the information about train timings, which we plan to get from the eRail API.

Having done this, calculating for overlap between journeys is nothing but the weighted average of overlap between different checkpoints. The weights for the averaging will be decided by the distance and transport preference fit of the overlapping journey parts. This is again, a well defined problem in its own right.