NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

Project Plan for TimeWise

Team 0
Lab group : SSP7

Group members	Roles
Mantri Raghav	Project Manager
Dwivedee Lakshyajeet	Development Lead
Harding James	Back-End Developer
Alex Bernini	Front-End Developer
Mittal Madhav	Release Manager
Xue Xueting	QA Engineer
Koh Hui Ling	QA Engineer
Lek Jie Ling	QA Manager

Version History

Version #	Implemented By	Revision Date	Reason
1.0	Raghav Mantri	20/02/2020	First version, completed sections 1,2,3. Started 4, 5
1.1	James Harding	03/03/2020	Sections 8,9,10 done
1.2	Hui Ling	09/03/2020	Sections 5, 6,7 done

Table of Contents

Introduction

Project Overview

The TimeWise app is a productivity tool for students which helps them import their school calendar as to-dos and allows them to additionally create their own tasks for each day.
The system will allow a user to login using their university credentials and then their timetable will be scraped from the university website and added to our database. They will then be given a 7 day overview of lessons that they have enrolled in and will be able to add personal events to this overview.

Project Description and Scope

TimeWise is being developed as a standalone utility app for NTU students with the aim of making daily and weekly planning of tasks and courses easier for students. The main users of this app will be NTU students. The existing planning apps will be researched thoroughly to understand their shortcomings. Most task planning apps don't contain the course schedule of the student which either forces the student to copy each class manually into the app or to use 2 different apps which is cumbersome. However, TimeWise will automatically download all course details and show it on the app, leaving the student with just the work of adding their own tasks. This will also be made easier by recommending 5 time slots for the student to do this task.

This document proposes a mobile app TimeWise, which will have the following features:

1. Importing a student's timetable automatically from STARS
2. Allowing the user to Create, Read, Update, and Delete their own tasks
3. Recommend 5 vacant 1-hour slots during the day in which the user can add a task
4. Send a notification to the user 30 minutes before their task/class is scheduled

The system shall include all necessary user interfaces for the above mentioned features as well as API endpoints in the server to manipulate the database.
An NTU email and password will be required to use the app since these credentials are required to scrape the STARS planner.

Project Organization

Team Structure

The following is the list of roles and the team members:

- Project Manager: Raghav Mantri
- Quality Assurance Manager: Lek Jie Ling
- Lead developer: Lakshyajeet Dwivedee
- Quality Assurance Engineer: Koh Hui Ling, Xue Xueting
- Front-end developer: Alex Bernini
- Back-end developer: James Sebastian Harding
- Release Manager: Mittal Madhav

Roles and Responsibilities

Project Manager: Raghav Mantri

- Oversees project progress
- Approves and executes project plan
- Assigns tasks and reports status of project to team members
- Manages and motivates team members
- Represents the team to the outside world

Quality Assurance Manager: Lek Jie Ling

- Oversees QA progress
- Approves and executes QA plan
- Assigns tasks and reports status of QA planning and execution to the QA engineers

Lead Developer: Dwivedee Lakshyajeet

- Oversees code and development progress
- Approves and executes the programming and front-end designs
- Assigns tasks and reports status of QA planning and execution to the QA engineers

Quality Assurance Engineer: Koh Hui Ling, Xue Xueting

- Ensures acceptable software quality
- Designs testing strategies
- Creates and manages test plan
- Verify software requirements
- Executes test procedures

Front-End Developer: Alex Bernini,

- Build the prototype on Figma, an online tool

- Implements product’s front-end/ User Interface using React Native
- Design User Interface
- Ensure stability and response time of the system meet the requirements
- Creates user manual

Back-End developers: James Sebastian Harding

- Responsible for coding the back-end in NodeJS
- Responsible for implementing the database in MongoDB
- Develops concepts and algorithms required for the general functioning of the app
- Make sure server deployed on Heroku works correctly
- Application reacts properly according to the commands given by the user

Release Manager: Mittal Madhav

- Managing, planning, scheduling and controlling the TimeWise app build through different stages and environments
- Required to check the testing and quality for release purposes

Team Communication

The communication process is as follows:

1. Bi-weekly scrum meetings are held on Tuesdays at 2PM
2. Group announcements are sent through WhatsApp
3. Telephonic discussions are held as necessary
4. GitHub Projects Board is used to communicate issues and code changes

Process Definition

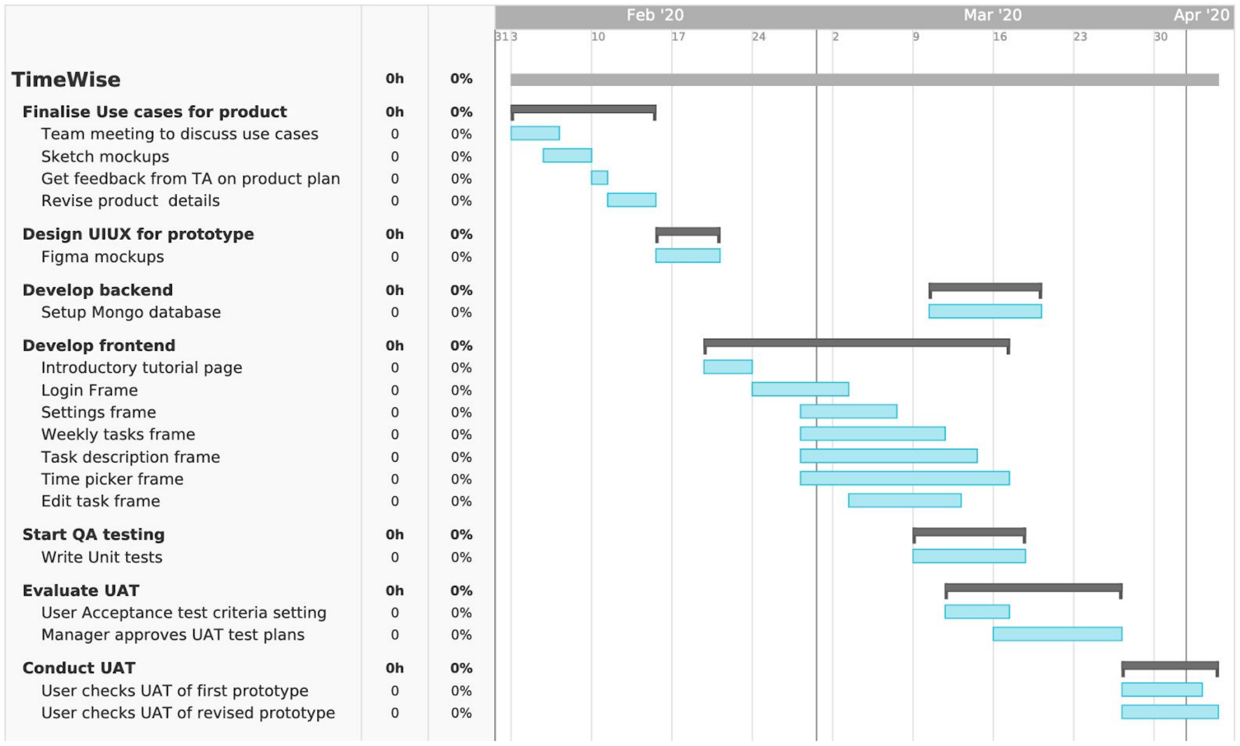
Lifecycle Model

Team 0 has chosen to adopt the Scrum model, which means that the project will progress via a series of sprints. Each sprint will be a sub-task of the entire project. During the sprint, the team will plan, build, test, and review a set of features. Each sprint will be 2 weeks long. At the end of each sprint, a sprint review will be conducted, to demonstrate the new functionality to the product owner. The product owner will then provide feedback on what could be done in the following sprint, to improve the product.
This will help in reducing the risks of :

- Producing a product that does not match the users' needs
- The project going over the time/budget constraints

Schedule

Activity Dependencies and Schedule



TimeWise	**0h**	**0%**
Finalise Use cases for product	**0h**	**0%**
Team meeting to discuss use cases	0	0%
Sketch mockups	0	0%
Get feedback from TA on product plan	0	0%
Revise product details	0	0%
Design UIUX for prototype	**0h**	**0%**
Figma mockups	0	0%
Develop backend	**0h**	**0%**
Setup Mongo database	0	0%
Develop frontend	**0h**	**0%**
Introductory tutorial page	0	0%
Login Frame	0	0%
Settings frame	0	0%
Weekly tasks frame	0	0%
Task description frame	0	0%
Time picker frame	0	0%
Edit task frame	0	0%
Start QA testing	**0h**	**0%**
Write Unit tests	0	0%
Evaluate UAT	**0h**	**0%**
User Acceptance test criteria setting	0	0%
Manager approves UAT test plans	0	0%
Conduct UAT	**0h**	**0%**
User checks UAT of first prototype	0	0%
User checks UAT of revised prototype	0	0%

Fig. 1 Gantt chart representing the schedule for the TimeWise project

Work Breakdown Structure

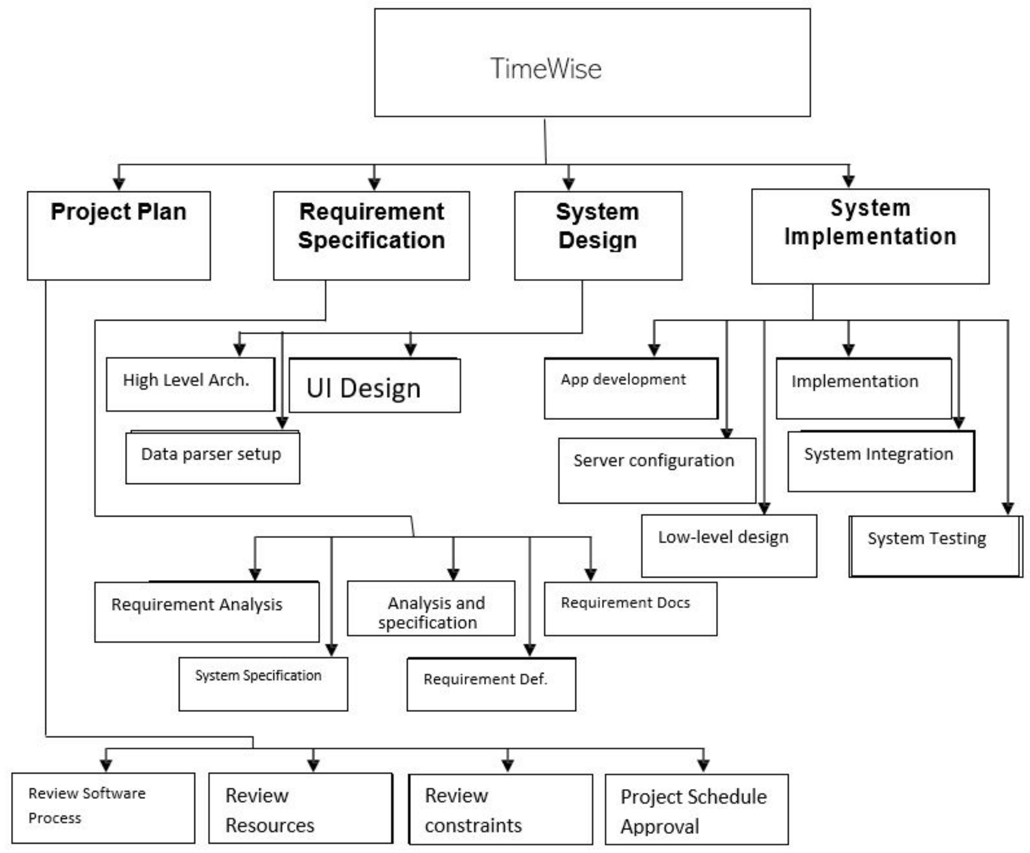


Fig. 2 Work breakdown structure for the TimeWise project

Work Packages

The entire project work is broken down by the important phases of the software development life cycle. They include the following:

1. Project Plan
2. Requirement Specification
3. User Interface
4. Technical Architecture
5. Coding & Unit Testing
6. Integration & Quality Assurance

Activity Dependencies

The following table describes the dependencies of the deliverable work packages:

Work Package #	Work Package Description	Duration	Dependencies
X01	Project Plan	18 days	--
X02	Requirement Specification	18 days	--
X03	User Interface	28 days	--
X04	Technical Architecture	15 days	X01,X02,X03
X05	Coding & Unit Testing	10 days	X04
X06	Integration and Quality assurance	26 days	X05

The following Activity Network Diagram describes the above in more graphical detail:

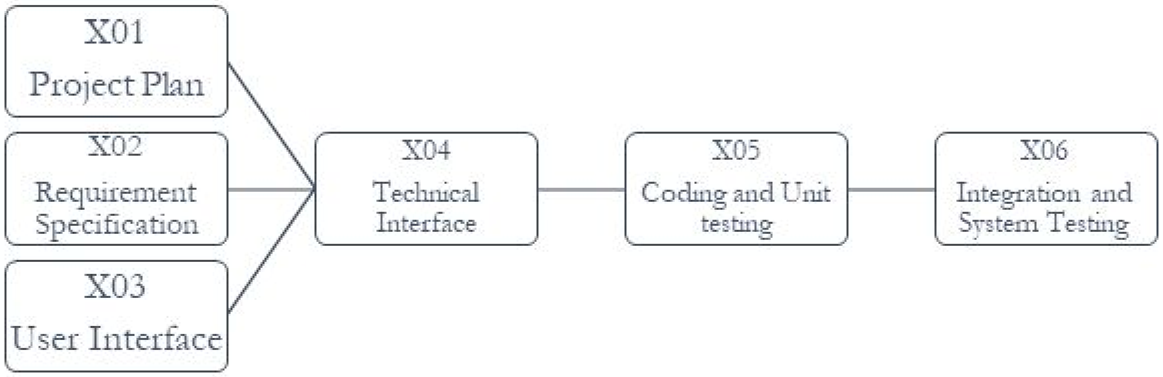


Fig. 2 Activity dependency diagram for TimeWise

Work Package Details

Work packages are listed below. A team member, indicated in bold, has been assigned as primarily responsible for each work package and will coordinate that package.

Project	TimeWise
Work Package	X01 - Project Plan (1 of 6)
Assigned to	Raghav Mantri, Hui Ling, James Harding
Effort	7 PD
Start Date	03/02/2020
Purpose	To determine an introductory overview of the project, to be refined in later work packages.
Inputs	None
Activities	This work package includes providing a brief overview of the project, its objectives, and a set of proposed project deliverables throughout the development of the software cycle. The people responsible for this work package will also be transcribing ideas brought up in the group meeting discussion into a formal report.
Outputs	A written document of the Project Plan introduction

Project	TimeWise
Work Package	X02— Requirement Specification (2 of 6)
Assigned to	Raghav Mantri, Hui Ling, James Harding
Effort	5 PD
Start Date	03/02/2020
Purpose	To establish a common understanding between the customer and the software project team of the customers' requirements to be addressed by the project
Inputs	None
Activities	Identify "the customer", interview customers, write and inspect customer requirements and build requirements.

Outputs	A written document of the requirement specification

Project	TimeWise
Work Package	X03— User Interface (3 of 6)
Assigned to	Alex Bernini, Madhav Mittal, Lakshyajeet Dwivedee
Effort	12 PD
Start Date	16/02/2020
Purpose	To build the user interface between the system and the customer, to make it easy use, and friendly to the customer
Inputs	User information
Activities	To get the user information, user request, display the dialog between system and user, display the result of request
Outputs	User Interface Mockups

Project	TimeWise
Work Package	X04— Technical Architecture (4 of 6)
Assigned to	Madhav Mittal, Lakshyajeet Dwivedee, James Harding
Effort	5 PD
Start Date	10/03/2020
Purpose	To do the high level architecture design
Inputs	Project Plan Work Packages (X01 to X03 inclusive).
Activities	High level design entails defining the architecture of the software system and identifying the various components and how they are inter-related to and interactive with each other. Designers also need to decide on the software and hardware infrastructures, such as what operating system on which the software is built, the language used to implement the software, and so on. Design topics including maintainability, portability, and reusability will be addressed here as well.
Outputs	High Level Design and Architectural Specification

Project	TimeWise
Work Package	X05 — Coding and Unit testing(5 of 6)
Assigned to	Madhav Mittal, Lakshyajeet Dwivedee, Alex Bernini, Lek, James Harding, Raghav Mantri, Xueting
Effort	14 PD
Start Date	09/03/2020
Purpose	To implement the system as per the requirements specification and other associated documents. This work package includes such additional activities as preliminary unit testing.
Inputs	Project Plan Work Package X06.
Activities	Programmers will implement the modules according to the design specifications noted in the Specification document. QA Engineers will create test cases and implement tests.
Outputs	Source code and testing files/test cases

Project	TimeWise
Work Package	X07 — Integration and System Testing(6 of 6)
Assigned to	Madhav Mittal, Raghav Mantri, Xueting, Huiling
Effort	10 PD
Start Date	12/03/2020
Purpose	To implement the system as per the requirements specification and other associated documents. This work package includes such additional activities as preliminary unit testing.
Inputs	Project Plan Work Package X06.
Activities	Programmers will integrate the modules according to the design specifications noted in the Specification document. QA Engineers will create test cases and implement tests.
Outputs	Source code and testing files/test cases

Project Estimates

Code Size Estimation using Function Points

We calculated unadjusted function points based on the complexity of functions provided by this system. Code size is then estimated by adjusted function point.

1. Unadjusted Function Points

TimeWise supports the following proposed functions:

- Login
- View/Modify/Delete list of classes for the next seven days
- Add/Modify/Delete list of tasks for the next seven days
- Send notifications for upcoming tasks
- Recommend five time slots
- Refresh schedule
- Logout

The measure of unadjusted function points is based on five primary component elements of these functions: Inputs, Outputs, Inquiries, Logical Files, and Interfaces. Each element ranges from Low Complexity, Medium Complexity to High Complexity. The detailed evaluation of the complexity is as follows:

Rating Inputs:

- Login Details(NTU email, password)
- University schedule(classes with time, day and name)
- Custom Tasks Details (time, day and name of event)

Files Type Referenced (FTR)	Data Elements		
	1-4	5-15	Greater than 15
Less than 2	Low (3)	Low (3)	Average (4)
2	Low (3)	Average (4)	High (6)
Greater than 2	Average (4)	High (6)	High (6)

Rating Outputs:

- Displaying the list of classes the user has (events with time, day and name)
- Displaying the list of user created tasks (time, day and name of event)

File Types Referenced (FTR)	Data Elements		
	1-5	6-19	Greater than 19

Less than 2	Low (4)	Low (4)	Average (5)
2 or 3	Low (4)	Average (5)	High (7)
Greater than 3	Average (5)	High (7)	High (7)

Rating Inquiries:

- Selecting the tasks and classes
- Selecting the settings

File Types Referenced (FTR)	**Data Elements**		
	1-5	6-19	Greater than 19
Less than 2	Low (3)	Low (3)	Average (4)
2 or 3	Low (3)	Average (4)	High (6)
Greater than 3	Average (4)	High (6)	High (6)

Rating Logical Files:

- Task Details
- Student Account

Record Element Types (RET)	**Data Elements**		
	1 to 19	20 - 50	51 or More
1 RET	Low (7)	Low(7)	Average (10)
2 to 5 RET	Low (7)	Average (10)	High (15)
6 or More RET	Average (10)	High (15)	High (15)

Rating Interfaces:

- 2 External Files Referenced (Tasks, Schedule)

Record Element Types (RET)	**Data Elements**		
	1 to 19	20 - 50	51 or More
1 RET	Low (7)	Low(7)	Average (10)
2 to 5 RET	Low (7)	Average (10)	High (15)
6 or More RET	Average (10)	High (15)	High (15)

Summary of above analysis:

Element	Complexity	Detail
Inputs	Low	Login Details
	Low	University Schedule
	High	Custom Task Details
Logical Files	High	Task Details
	Medium	Student Account
Outputs	High	Display University Timetable
	High	Display User Created Tasks
Inquiries	High	Selecting the tasks and classes
	Low	Selecting the settings
Interfaces	Medium	Tasks, Schedule

Calculation of Unadjusted Function Points:

Characteristic	Low		Medium		High	
Inputs	2	× 3	0	× 4	1	× 6
Outputs	0	× 4	0	× 5	2	× 7
Inquiries	1	× 3	0	× 4	1	× 6
Logical Files	0	× 7	1	× 10	1	× 15
Interfaces	0	× 5	1	× 7	0	× 10
Unadjusted FP	9		17		41	
Total=L+M+H	**67**					

2. Adjusted Function Points

Influence Factors	Score	Detail
Data Communications	0	No data and/or control information would be sent or received over communication facilities.
Distributed Functions	4	Distributed processing and data transfer are online and in both directions.
Performance	3	Response time or throughput is critical. No special design for CPU utilization was required.
Heavily used	2	Some security or timing considerations are included.
Transaction rate	3	Daily peak transaction period is anticipated.
On-line data entry	1	Some transactions are interactive data entry
End-user efficiency	2	Four to five of the efficiency designs are included
On-line data update	3	Online update of major internal logical files is included.

Complex processing	0	None.
Reusability	4	The application was specifically packaged and/or documented to ease re-use, and the application is customized by the user at source code level.
Installation Ease	1	Installation is easy, trouble-free.
Operational Ease	1	Effective start-up, back-up, and recovery processes were provided, but no operator intervention is required (count as two items).
Multiple sites	0	User requirements do not require considering the needs of more than one user/installation site.
Facilitate change	3	Flexible query and report facility is provided that can handle complex requests, for example, *and/or* logic combinations on one or more internal logical files (count as three items).
Total score	27	
Influence Multiplier = Total score × 0.01 + 0.65 = 27 × 0.01 + 0.65 = **0.92**		
Adjusted FP = Unadjusted FP × Influence Multiplier = 67 × 0.92 = **61.64**		

Scoring (0 – 5)
0 = No influence
1 = Insignificant influence
2 = Moderate influence
3 = Average influence
4 = Significant influence
5 = Strong influence

3. Lines of Code

According to Capers Jones statistics, each Function Point requires 47 lines of code if the application is implemented. Therefore, we have:

Lines of Code = 61.64 FP × 47 LOC/FP = **2897 LOC**

Efforts, Duration and Team Size Estimation

To estimate the effort and duration required for the project, we use function points as the basis to calculate Effort, Duration, Team size and finally the schedule. The estimates are expanded to account for project management and extra contingency time to obtain the total average effort estimates. From these averages, the duration of each work package in working days is estimated based on the following calculations.

- Working days include 5 days in a week.

- Effort = Size / Production Rate = (2897 LOC) / (39 LOC/PD)[1] = 74 PD
- Duration = 3 × (Effort) 1/3 = 3 × (74) 1/3 = 12.6 Days
- Initial schedule = 12.6 Days / 5 days a week = 2.52 Weeks
- Team size = 74 PD / 12.6 D = 5.87 P = 6 Persons
- Working hours include 8 hours in a working day.
- Total person-hours (PH) = 74 PD × 8 hours = 592 PH

1. Distribution of Effort

1990's Industry Data	Work Package	Distribution	Estimates
Preliminary Design 18 %	Project Plan	9%	53.28
	Requirement Specification	9%	53.28
Detailed Design 25 %	User Interface	7%	41.44
	Technical Architecture	11%	65.12
	Data Modeling	7%	41.44
Code & Unit Testing 26 %	Code & Unit testing	21%	124.32
	Online Documentation	5%	29.6
Integration & Test 31 %	Integration & Quality Assurance	31%	183.52
	Extrapolated total effort		**592**
	2% for project management		11.84
	3% for contingency		17.76
	Total effort		**621.6**

These duration estimates are based on the assumption that each team member works an equal amount on any given work package.

[1] Lines of code per Person Day statistics based on Industrial Benchmarks, 1997: 31 LOC/PD for United States; 62 LOC/PD for Canada

Cost Estimates

Item	Supplier	Quantity	Unit Price	Total
Project Manager		1	$30,000	**$30,000**
Project Team Members		7	$3,000	**$21,000**
Computers	Dell	8	$1,000	**$8,000**

Publication License	Google, Apple	2	$100	**$200**
Server	Heroku	1	$2,800	**$2,800**
Database	MLab	1	$15,000	**$15,000**
Office Rent	NTU	1	$6,000	**$6,000**
			Total	**$83,000**

TimeWise is not responsible in any way for supplying said systems. TimeWise's hardware and software responsibilities relate only to our own development needs to accomplish the project we have been asked to complete, and which has been described in the introduction section of this document. TimeWise will also demonstrate the completed product.

Product Checklist

The plan is that the items listed below will be delivered on the stated deadlines.

Project Deliverable	**Estimated Deadline**
Requirements Specification	Feb 20th, 2020
Project Plan	Mar 12th , 2020
Module/System Test Plan	Mar 26th, 2020
System Release (Demo)	Mar 26th , 2020

Best Practice Checklist

Practice
Need to document what we do; all documentation must be in a standardized format.
Pay attention to requirements, check for ambiguity, completeness, accuracy, and consistency. The requirement documentation must contain a complete functional specification for the application.
Keep it simple. Complexity management is one of the major challenges. Strive to: • Minimize interfaces between modules, procedures and data. • Minimize interfaces between people, otherwise exponential communication cost • Avoid fancy product functions, design as long as the functionality meets the customer requirements, meaning it must be user friendly
Require Visibility. We must see what we build or else we can measure the progress and take management action. This would mean that the team leader must have good communication with his or her team members, require developers to make code available for review and also review design for appropriateness.
Plan for continuous change. We must: • All manuals designs, test, source code should have revision numbers and dates revision history comments, change marks to indicate the changes made • New revisions should be approved before being made and checked for quality and compliance after being made
Do not underestimate. We must be careful to obtain accurate estimates for: time, effort, overhead, meeting time, and especially effort on integration, testing, documentation and maintenance.
Code reviews are a much more efficient method to find software defects. Plan and manage code reviews between team members.
Software testing will use both black box and white box testing. It will involve unit, functional, integrating and acceptance testing.

Risk Management

Besides the general risk management, the following risks have been identified for the TimeWise project:

Changes to requirements:
Impact Severity: High
Probability: 25%
Impacts: Depending on the stage at which changes occur, could range from needing to update the requirements documentation to a needed complete redesign.
Risk Reduction: Be rigorous in eliciting requirements. Make stakeholders aware of potential repercussions of requirement changes.

Specification Delays
Impact Severity:High
Probability: 15%
Impacts: Delay in finalizing the specification will push the schedule for all following stages of the project.
Risk Reduction: Monitor progress of specification carefully.

System size underestimated
Impact Severity: Moderate
Probability: 30%
Impacts: More work will need to be spent on design and coding; could negatively impact schedule.
Risk Reduction: Update estimates if needed as the project progresses.

Staff leaving before project complete
Impact Severity: Extreme
Probability: 5%
Impacts: There would be more work for remaining employees, and any specialized skills or knowledge would be lost.
Risk Reduction: Offer benefits and incentives to staff.

Problems coordinating within group
Impact Severity: Moderate
Probability: 40%
Impacts: Members may be unaware of what is expected of them; managers may not be able to measure progress; portions of projects not completed.
Risk Reduction: Follow communication plans as documented in section 2.3

Customer cancels project
Impact Severity: Extreme
Probability: 1%
Impacts: All work will have been wasted.

Risk Reduction: Keep in close contact with the customer ensuring that they have completed some market research indicating a demand for this product.

Lack of staff motivation
Impact Severity: Modorate
Probability: 20%
Impacts: Some staff members may not be as motivated as others due to either the environment or lack of interest in the project.
Risk Reduction: Try to keep development informal and interesting as well as offering incentives to staff members.

Quality Assurance

The project will achieve quality assurance by following the standards set by us. The specific procedures and details are provided in the Quality Plan but involve testing the system so as to determine that the software is of sufficient quality. Specific test procedures and details shall be provided in the Module/System Test Plan.

The team shall make use of two test methodologies:

1. Unit Testing - A form of testing which involves testing individual components separately
2. Integration Testing - A form of testing which involves testing the behaviour of different units when they are combined with each other
3. System Testing - Testing of entire system by independent testers
4. User Acceptance Testing - Gathering feedback from prospective end-users

We will make broad use of realistic test cases. Detailed test data is an important part of the final project delivery and will be used to determine if the product is of sufficient quality. The Timewise client is expected to display and enter data regarding times, dates and names of events and users, it shall be provided a comprehensive and detailed subset of this data for testing purposes. The data will be from members timetables. As such we will be using realistic test data to validate our code and results. In addition, extreme cases (such as when a user logs in but has no timetabled events) will be used to ensure that the system behaves correctly even in degenerate cases.

Monitoring & Control

Many procedures are required in order to be able to successfully monitor the progress of the software project. Some of the most important are:

Quantitative measurement of resource consumption: Estimates on the TimeWise projects resource requirements, primarily in terms of human resources, can provide a quantitative measurement of project progress when compared to progress in terms of project milestones. The percentage estimates of each milestone's resource requirements provided in this document allow for easy progress tracking.

Identification of major project risks: Early identification of major risks to the project allows for placement of preventative measures before problems can develop. Major risks have been identified in the Risk Management section of this document, along with the measures being taken to avoid them.

Regular reviews of project progress: Throughout the duration of the TimeWise project, the team shall meet weekly to review the progress of all project tasks, including management, planning, analysis, development, and testing.

Timeline Planning and task decomposition: This document outlines an estimated timeline for the project. A reasonably accurate timeline can be assembled by hierarchically decomposing tasks into measurable subcomponents and estimating requirements for each. At the same time, this decomposition can assist in task assignment and balancing. Throughout the implementation phase, these subcomponents can allow for fine-grained measurement of progress. Project subcomponents and timeline estimates are included in the Estimates and Work Breakdown Structure sections of this document.