# Chapter-1 Introduction

## Project introduction

Tdolist & Note Manager is a web application in which users will be able to note down notes and plan the activities. This application provides area to users to plan activities and check which are left to be complete. Here users also able to keep note important note and can share it.

## Justification to the project

To solve the problems that every notebook has which is availability. This web application is introduced to solve that problem and to provide extra features to the users.

## Background of the project

## Problem statement

## Description of the project

Todolist & Note Manager is an online notebook and activity checking application that can be access from anywhere. This application provides space to users for keeping users notes and providing access to their note anytime.

## Features

* Easy access of notes anytime
* Change styles easy and quick
* Pre-installed templets of notes
* Share notes
* Search notes
* Group to do list and planning
* Backup notes
* Change background color of note

## Overview of the project

Todolist & Note Manager is a web application where users can create notes, share and make activities plan.

# Chapter-2 Scope of the project

## Scope

## Limitation

This application has many merits point but also some limitation of application. Some are listed:

1. To have access towards notes one must have internet connection.
2. This application does not supports audio or video for attachment.

## Aims

* To help users to create and save notes.
* To helps users to track completion of their activities.

## Objectives

* Organize all notes in one place.
* Share notes to co-workers and friends.

## Overview

This web application is introduced for users to create and manipulate the created notes. This application another main agenda is to make list of activities to be done and what left to be done (to do list).

# Chapter-3 Development methodology

## Description of methodology

**Agile software development**

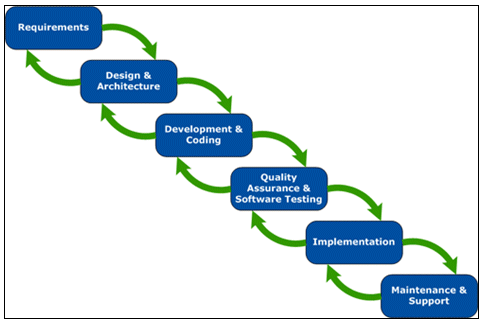
Agile is a software development methodology to software development. Agile is collaboration of all team members and clients/users. It is a process of doing a particular task by dividing into many chumps of tasks, which has to be completed in certain period and repetitions of same cycle until task is complete.

Figure 1Agile development life cycle

**Benefits of Agile software development**

* Satisfaction and high involvement of stakeholder
* Transparency towards clients
* Overall cost and schedule prediction
* Can change requirements if needed
* More focus on users and high quality of product due to involvement of clients

**Waterfall model**

Waterfall model is s also referred to as a linear-sequential life cycle model. It is very simple to understand and use. In a waterfall model, each phase must be completed before the next phase can begin and there is no overlapping in the phases. This model contains following phases:

1. Requirement gathering and documentation
2. System design
3. Implementation
4. Testing
5. Deployment
6. Maintenance

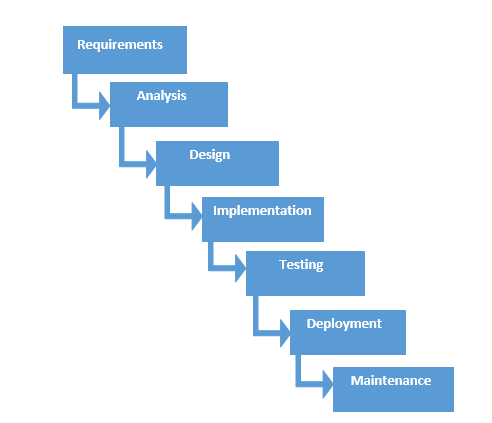


Figure 2 Waterfall model

**Benefits of Waterfall method**

* This method is simple and easy to understand
* The requirements of project will be unchanged which helps to make the products that we want to build
* Every steps must be complete to proceed further
* Steps do not overlap each other
* Start date and end date of every steps can be identified which helps to easy to measure progress

## Design pattern

**Model View Controller (MVC)**

Model View Controller is a common architectural pattern for developing software. MVC divides application into three main parts that are interconnected. It contains data model, presentation layer and control layer.

Figure 3 Model view controller design pattern

**Abstract factory**

Abstract factory pattern is also known as factory of factories. In this design pattern creates factory which creates other factories. Abstract factory falls under creational pattern. Here an interface creats factory of related objects without explicitly identifying their classes.

Figure 4Abstract factory desing pattern

**Factory method**

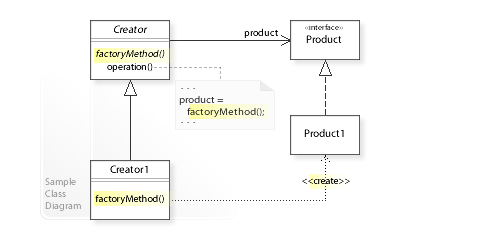
Factory method design pattern also falls under creational pattern. Here we create object without showing the creational logic to the end user and refer to newly created object using common interface.

Figure 5Factory method

After analyzing some different design pattern, I prefer MVC design pattern for this application.

**Architecture**

# Chapter-4 Work Breakdown Structure

## Work Breakdown Structure (WBS)

Work Breakdown Structure is breakdown of project into small parts. WBS helps to achieve project objectives. WBS provides framework for detailed cost estimation and schedule development. It is tree structure that display subdivision of that main part.

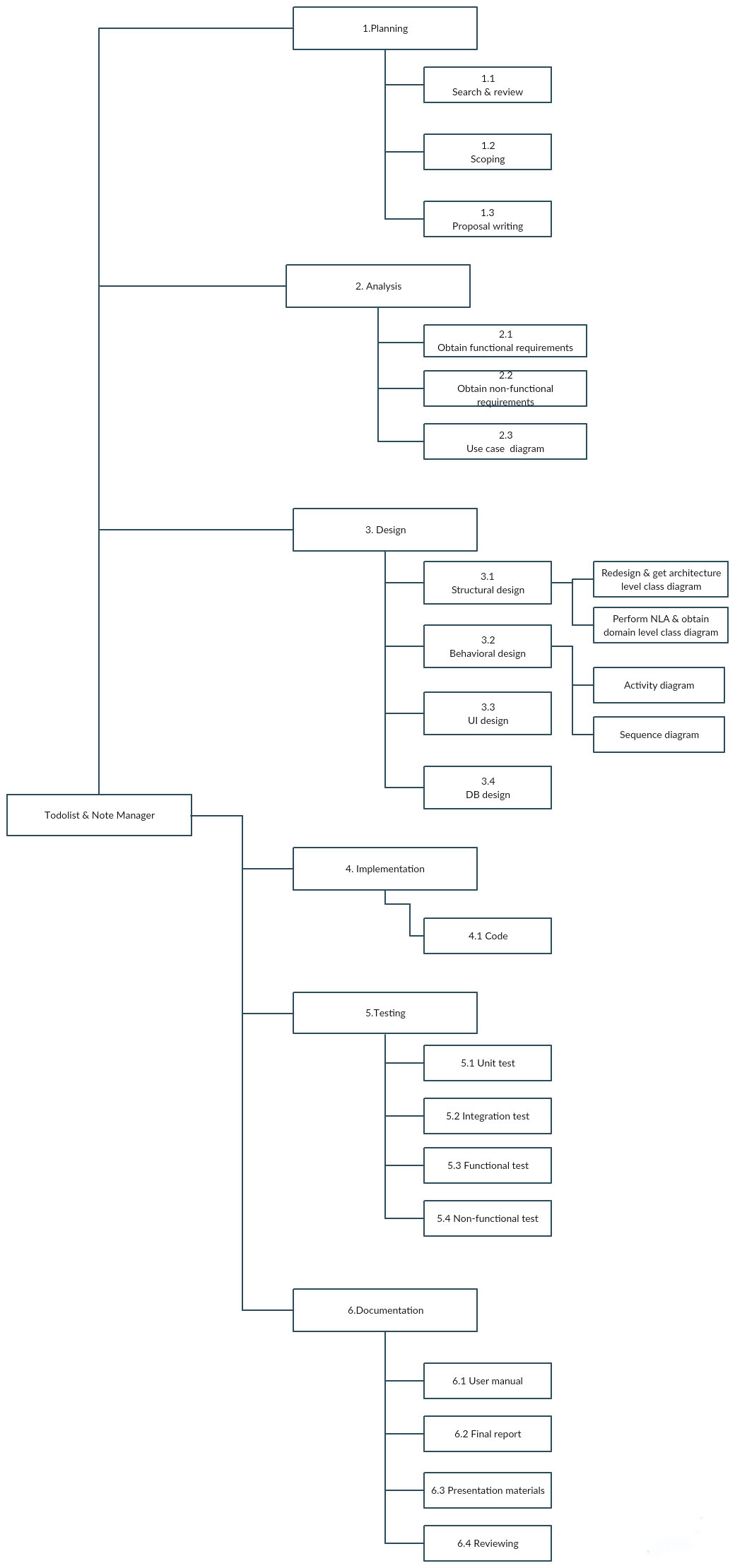


Figure 6 WBS

|  |  |  |
| --- | --- | --- |
|  | **Task name** | **Time estimated(days)** |
| 0 | **Todolist & Note manager** |  |
| 1 | **Planning** |  |
| 1.1 | Search & review |  |
| 1.2 | Scoping |  |
| 1.3 | Proposal writing |  |
| 2 | **Analysis** |  |
| 2.1 | Obtain functional requirements |  |
| 2.2 | Obtain non-functional requirements |  |
| 2.3 | Use case diagram |  |
| 3 | **Design** |  |
| 3.1 | Structural design |  |
| 3.1.1 | Redesign & get architecture level class diagram |  |
| 3.1.2 | Perform NLA & obtain domain level class diagram |  |
| 3.2 | Behavioral design |  |
| 3.2.1 | Activity diagram |  |
| 3.2.2 | Sequence diagram |  |
| 3.3 | UI design |  |
| 3.4 | DB design |  |
| 4 | **Implementation** |  |
| 4.1 | Code |  |
| 5 | **Testing** |  |
| 5.1 | Unit test |  |
| 5.2 | Integration test |  |
| 5.3 | Functional test |  |
| 5.4 | Non-functional test |  |
| 6 | **Documentation** |  |
| 6.1 | User manual |  |
| 6.2 | Final report |  |
| 6.3 | Presentation material |  |
| 6.4 | Reviewing |  |
|  |  |  |
|  |  |  |

## Milestone