Project Plan

for

WikiMon

E12

Name	**SRN**
Vishwa Pravin	PES1201800143
Anup N	PES1201800283
Sreesh	PES1201801580

Contents

1 Introduction

1.1 What this document covers

This document outlines the project plan for WikiMon an open-collaborative knowledge platform.

Sections 2, 3 covers the basic workflow and neccessary tools to be used for this project, *Section 4* covers the required functionalites to be completed, more functionalites will be added or removed as the project proceeds to completion.

Section 5 onwards discusses the deliverables, how important they are and an inceptive timeline for this project.

2 Lifecycle - Agile

The most important feature of knowledge repository is updating and continuous integration of the latest data and deprecating the older facts. Continuous integration and continuous development is the foundation of the Agile approach. Agile model is a combination of iterative and incremental process models with focus on process adaptability and customer satisfaction by rapid delivery of working software products.

Agile Methods break the product into small incremental builds. These builds are provided in iterations. Each iteration typically lasts for a short span of time. Iterative approach is taken and working software build is delivered after each iteration. Each build is incremental in terms of features; the final build holds all the features required by the customer. At the end of the iteration, a working product is displayed to the customer and important stakeholders.

Every detail presented by WikiMon is fully reviewed and rewritten until it is flawless and users regularly suggest edits, hence continuous integration and continuous deployment is a very essential tool for a knowledge repository. Since agile development is focused on quick responses to change and continuous deployment, while providing flexibility to the users is most ideal for projects like knowledge repository(WikiMon).

While other software development life cycles can be used, considering the size of the project to be small and the necessity of features of continuous development and continuous integration we can say that an agile development enables the product to adapt to user's edit suggestion and reviews quickly and efficiently. So as to ensure a quick implementation of the project.

3 Tools

3.1 Languages/Frameworks

React: The front end javascript library React is used to build and maintain our User Interfaces (or UI) components. It is one of the most popular and widely used libraries for various frontend tasks.

Flask: This is the python framework we use for developing our web applications since it is capable of RESTful request dispatching and is easily deployable in production. PostgreSQL: For satisfying the database requirement, we use the open source object relational database management system (ORDBMS) PostgreSQL due to its robustness.

Nginx: The web server we use for serving our website is the open source software nginx. We use this due to its lightweight resource utilization and scalability. It also comes in handy for load balancing.

PostgreSQL For satisfying the database requirement, we use the open source object relational database management system (ORDBMS) PostgreSQL due to its robustness.

Markdown: To display the content on our webpage, we use Markdown. It allows us to write information using an easy-to-read and easy-to-write text format

3.2 Storage/services

AWS S3: To store our data, we use Amazon's Simple Storage Service (Amazon S3). It is a simple web service interface that we can use to store and retrieve any amount of data at any time from the web.

AWS EC2: To host our website we use Amazon's Elastic Cloud Computing (Amazon EC2). Amazon EC2 provides resizable compute capacity in the cloud. It is designed to make web-scale cloud computing easier for developers and allows maximum scalability and availability for websites and web applications hence paying only for what we use.

3.3 Code editing/version control

Vscode editor: For editing and debugging of our code, we use Visual Studio Code editor since it has features such as syntax highlighting, intelligent code completion, snippets, code refactoring.

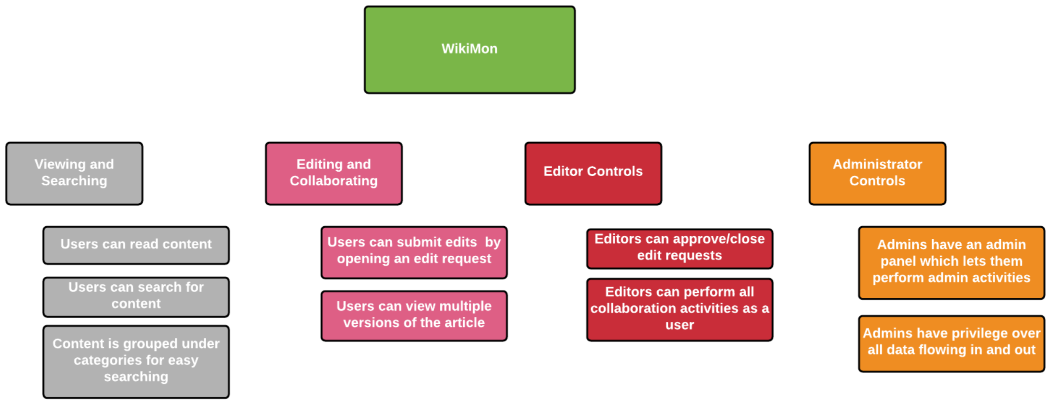
Github, Git: For version control of our code, we make use of Git since it the most popular tool used for version controlling and for storing copies of our code on the web, we upload it to Github by creating a repository and uploading code on a regular basis until completion.

Chrome: To view the frontend (the end which the user sees) of our designed website, we use the Google Chrome web browser. Chrome is preferred due to the fast loading speed of websites.

Windows/Linux: All of the softwares is run on the Windows or the Linux operating systems

4 Functionalities

4.1 Overview


WikiMon
Viewing and Searching
Editing and Collaborating
Editor Controls
Administrator Controls
Users can read content
Users can search for content
Content is grouped under categories for easy searching
Users can submit edits by opening an edit request
Users can view multiple versions of the article
Editors can approve/close edit requests
Editors can perform all collaboration activities as a user
Admins have an admin panel which lets them perform admin activities
Admins have privilege over all data flowing in and out

The above diagram shows a deliverable oriented work breakdown structure of the functionalities, any additional details is mentioned in the following sections.

4.2 Viewing and Searching

4.2.1 Reading content

Users can read content in their web browser.

4.2.2 Searching for content

Users can search for content by specifying a part of the title or search with categories.

4.2.3 Categories

All content is placed under categories. Categories themselves can be a part of another category.

4.3 Editing and Collaborating

4.3.1 Edit Requests

Users can open edit requests for changing content which can be viewed by everyone.

4.3.2 Viewing multiple versions

Users can view previous versions of the same content and can view them.

4.4 Editor Controls

4.4.1 Approving Edits

Editors can approve them, suggest more edits, edit it themselves, or delete them. An edit requires a minimum number of approvals before being submitted.

4.4.2 Editor Collaboration

Editors can themselves open edit requests just like users.

4.5 Adminstrator Controls

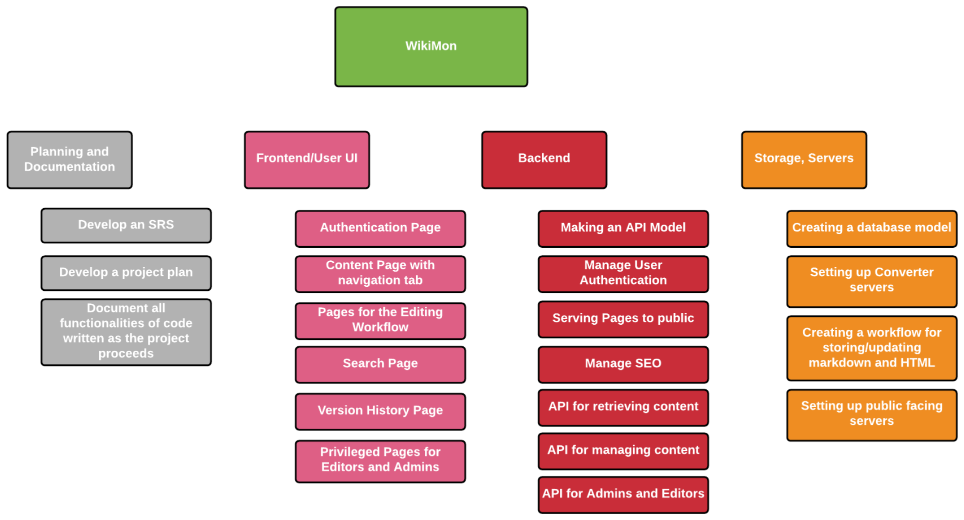
4.5.1 Admin Panel

Admins have an admin panel which lets them monitor all activity

4.5.2 Viewing all data

Admins can view data which is not available to editors and users.

5 Deliverables


WikiMon
Planning and Documentation
Develop an SRS
Develop a project plan
Document all functionalities of code written as the project proceeds
Frontend/User UI
Authentication Page
Content Page with navigation tab
Pages for the Editing Workflow
Search Page
Version History Page
Privileged Pages for Editors and Admins
Backend
Making an API Model
Manage User Authentication
Serving Pages to public
Manage SEO
API for retrieving content
API for managing content
API for Admins and Editors
Storage, Servers
Creating a database model
Setting up Converter servers
Creating a workflow for storing/updating markdown and HTML
Setting up public facing servers

The above image shows a work breakdown structure of the deliverables. More information about each deliverable is in the following sections.

All the deliverables will be completed parallelly, for instance documentation will keep changing as the project is built.

5.1 Overview of Deliverables

Deliverable	Any Reuse: Additional Details	Time and Effort
Planning and Documentation		
Develop an SRS	Reusing existing models wherever required	1 month
Develop an Project Plan	Reuse existing plan structures wherever required	1 month
Develop an Project Plan	Reuse existing plan structures wherever required	1 month
Develop Documentation	Use existing documentation for any product used where necessary	3 months

	Frontend/User UI	
Frontend UI	React and additional addons	-
Authentication Page	Using google oauth workflow	1 month
Content Page	-	3 weeks
Editing Workflow	-	1 month
Search Page	-	2 months
Version History Page	-	2 month
Privileged Pages	-	2 month
	Backend	
Making an API Model	-	1.5 month
Manage User Authentication	-	3 weeks
Serving Pages to public	-	1.5 month
Manage SEO	Using google analytics	1 month
API for retrieving content	-	1.5 months
API for managing content	-	1 month
API for Admins and Editors	-	2 month
	Storage/Servers	
Creating a database model	-	1 month
Setting up Convertor servers	-	2.5 month

Workflow for storing/updating	-	2.5 month
Setting up public facing servers	-	1 month

5.2 Planning and Documentation

This section covers all the documentation processes for the project

5.2.1 Developing an SRS

An initial SRS will be created and the project will follow as per its requirements, if any changes are made midway then the SRS will be updated to a newer version.

All the documents will be built from ground up, any extra software documentation may be copied here and linked to wherever needed.

5.2.2 Developing a Project Plan

An initial project plan will be created in sync with the SRS and will be maintained as the project is built.

5.2.3 Documentation of Code

As code is written a seperate slot of time will be dedicated towards adding documentation for that code. Keeping the documentation up to date will ensure the code will be reusable by multiple teams without hassle.

5.3 Frontend

This section covers the User Interface for the project, the frontend targets the web platform and users the React framework.

5.3.1 Authentication Page

This page will handle authentication of the user, the authentication scheme will support password, google oauth as decribed in the SRS.

5.3.2 Content Page

The content for any article will be displayed here, the interface contains a navigation sidetab, the content section and citations in the end.

Users will view the HTML form of the content in the browser, which will be created from the source markdown in which the content was written.

5.3.3 Pages for Editing Workflow

As described in the SRS the editing workflow consists of

- Editing - Users can change the markdown of the content
- Preview - Users can view the chnages they made
- Submit - User can open a edit request for their changes

5.3.4 Search Page

In this page users will have a search box along with additional filters for categories, users can search and navigate from here.

Google search addon will be looked into if feasible.

5.3.5 Version History Page

All versions of the content will be listed here, users can view any of them.

5.3.6 Privileged Pages

This is for additional pages for both editors and admins, Admins will have a dashboard to see all data and traffic coming in and going out, Pages for editors and more detailed information is not yet decided for the inital project plan.

5.4 Backend

This section covers deliverables for the backend, the backend uses python+flask framework running with a nginx server. All communication is performed using REST APIs.

5.4.1 Making an API Model

The API Model for interaction between users,editors, admins and the interaction with servers for converting markdown to html and maintaining versions will be modeled.

5.4.2 Manage User Authentication

APIs for user auth will be implemented as part of this deliverable.

5.4.3 Serving pages to public

Nginx servers will be configured to serve pages to public, using caches will also be looked into as the project is built. This is a part of *storage, servers* section.

5.4.4 Manage SEO

Setting up pages to make sure SEO ranking, this will be started once the project is built and data about page loads will be available.

5.4.5 API for retrieving content

APIs for retrieving content will be implemented as part of this deliverable.

5.4.6 API for managing content

APIs for managing content will be implemented as part of this deliverable.Users can submit edit requests, editors and admins will have privileges routes to manage content in an internal network.

The management apis for admins and editors is a part of the next section.

5.4.7 API for Admins an Editors

APIs for Admins an Editors will be implemented as part of this deliverable.

5.5 Storage, Servers

This sections covers infrastructure, storage related items which includes public facing servers, private markdown converter servers, databases, and content storage system. The servers are hosted on AWS EC2 instances, postgresql database will be used.

During development, content storage will be handmade version control system will be used by having multiple copies of the file. Later git will be used if feasible. The version control and its interface will be seperate so that different verions control system can be swapped.

5.5.1 Creating a database model

A database model will be created for all entities users, content, logging.

5.5.2 Setting up Convertor servers

Convertor servers convert markdown to html. Pandoc running on linux will be used for this process.

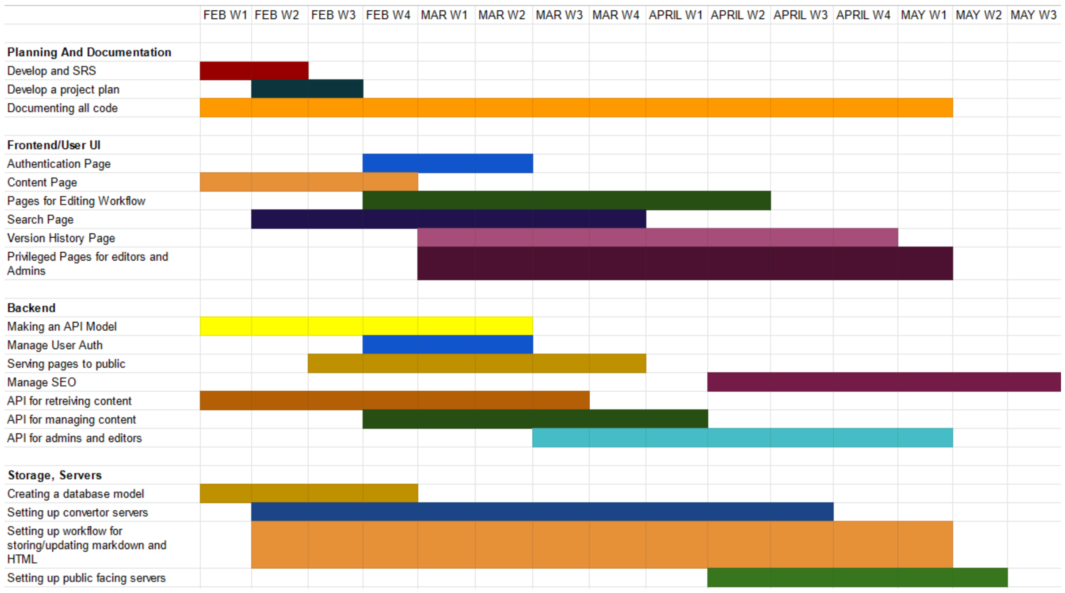
5.5.3 Workflow for storage and updates

As said in the section introduction, a version control system will be used to store and a interface will be developed which will be abstracted from the version control system. This interface will be used for storage and updates. The accurate workflow can be any as long as it is feasible.

5.5.4 Setting up public facing servers

This project will fully reside on the cloud using AWS. Hence public facing servers will be hosted using EC2 instances. During development a single instance will be used while keeping code as scalable as possible. Once complete scalabilty will be looked into, which might demand docker, kubernetes.

6 Timeline



	FEB W1	FEB W2	FEB W3	FEB W4	MAR W1	MAR W2	MAR W3	MAR W4	APRIL W1	APRIL W2	APRIL W3	APRIL W4	MAY W1	MAY W2	MAY W3
Planning And Documentation															
Develop and SRS	■	■													
Develop a project plan		■	■												
Documenting all code	■	■	■	■	■	■	■	■	■	■	■	■	■		
Frontend/User UI															
Authentication Page				■	■	■									
Content Page	■	■	■	■											
Pages for Editing Workflow				■	■	■	■	■	■	■					
Search Page		■	■	■	■	■	■	■							
Version History Page					■	■	■	■	■	■	■	■			
Privileged Pages for editors and Admins					■	■	■	■	■	■	■	■	■		
Backend															
Making an API Model	■	■	■	■	■	■									
Manage User Auth				■	■	■									
Serving pages to public			■	■	■	■	■	■							
Manage SEO										■	■	■	■	■	■
API for retreiving content	■	■	■	■	■	■	■								
API for managing content				■	■	■	■	■	■						
API for admins and editors							■	■	■	■	■	■	■		
Storage, Servers															
Creating a database model	■	■	■	■											
Setting up convertor servers		■	■	■	■	■	■	■	■	■	■				
Setting up workflow for storing/updating markdown and HTML		■	■	■	■	■	■	■	■	■	■	■	■		
Setting up public facing servers										■	■	■	■	■	