SCHOOL MANAGEMENT SYSTEM –FUNCTIONALITY ADDITION

PROJECT PLAN

Date	11 February 2019
Version	2.1
Date of Authorisation:	11 February 2019
Project start date	11 February 2019
Projected finish date	18 March 2019
Project Manager	Kephas Shiweda
Project Manager Contact Details	Mobile - +264 81 613 7758 Email - 'keshmabofele@gmail.com'
Project Members (alphabetical)	Dan Ntwari 216051533 Erich Godenschweig 216094518 Fikameni Petrus 211009725 Kephas Shiweda 211035750

Table of contents

Table of Contents

1. SCOPE STATEMENT

We have identified a school management system (SMS) that manages school related data such as, information for teachers, students, results, subjects and the administration of these elements.

Our aim in this project is to add functionality to the school management system by placing ourselves in the shoes of the users. The key functionalities that we see a need for are reports on the dashboard which provide content based on the user's access rights. In addition to this the reports should be easy to share and print.

1.1. PROJECT JUSTIFICATION AND DEFINITION

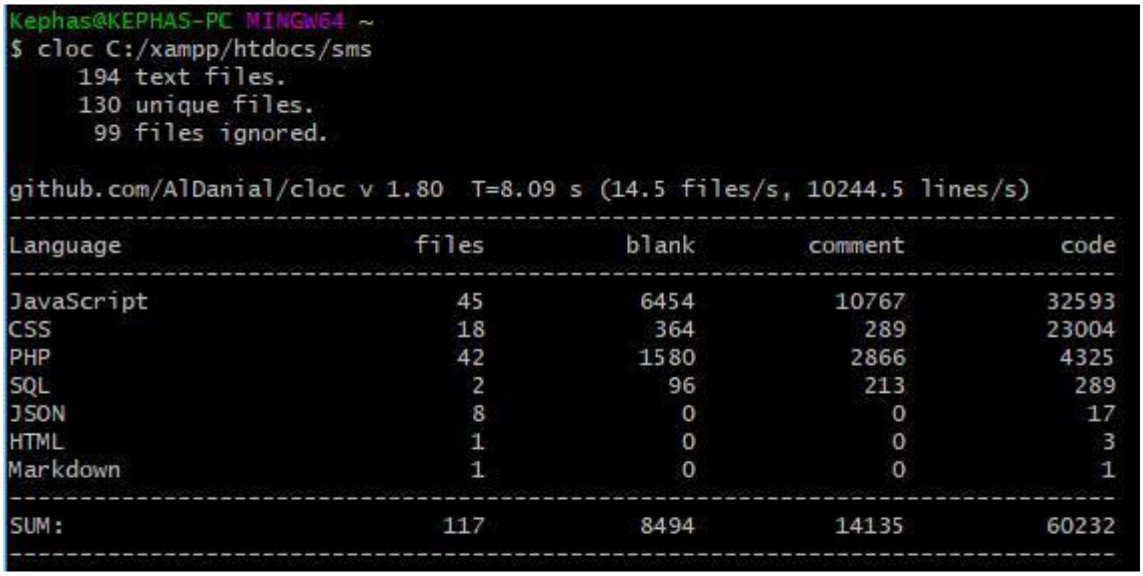
The aim of the project is to improve an open source software by doing so we found an Open source School Management System which is a cross-platform web based system, design by W3LAYOUTS and developed by Ravi Khadka. The SMS is 100% Dynamic with core PHP and it uses mysql database as a data storage facility. SMS is an information system to manage school related data/information. Different users such as students, teachers, student's parents, and system administrator are among the intended market. The whole purpose of the SMS is to design a generalized solution for an education system that is applicable to a broad range of schools and has support for administrative staff by generating quick summaries for decision making. The improvements will facilitating user experience and will allow for communication to the database in an automated way. The team will use an agile software development approach, where short 10 minute daily meetings will be held to gauge the progress of the project.

GitHub repository for project: https://github.com/KephasT100/sms

1.2. Software Stack

- JavaScript 71.6% - A web typescript tool that enables user interaction.
- CSS 22.2% - A styling and design component of web based programming.
- PHP 6.2% - A backend programming language that enables the retrieval and use of stored data.

Figure 1 - Lines of Code Count shows a total of more than 60 000 lines of code in the SMS project which makes it a very challenging and interesting project to work on.



```
Kephas@KEPHAS-PC MINGW64 ~
$ cloc C:/xampp/htdocs/sms
     194 text files.
     130 unique files.
      99 files ignored.

github.com/AlDanial/cloc v 1.80  T=8.09 s (14.5 files/s, 10244.5 lines/s)
-------------------------------------------------------------------------------
Language                     files          blank        comment           code
-------------------------------------------------------------------------------
JavaScript                      45           6454          10767          32593
CSS                             18            364            289          23004
PHP                             42           1580           2866           4325
SQL                              2             96            213            289
JSON                             8              0              0             17
HTML                             1              0              0              3
Markdown                         1              0              0              1
-------------------------------------------------------------------------------
SUM:                           117           8494          14135          60232
-------------------------------------------------------------------------------
```

Figure 1 - Lines of Code Count

2. TARGET GROUP (CLIENTS)

2.1.1. Students

Student Information such as grades, marks, and almost all the information regarding the students can be accessed easily using school management system. They will also be able to login to view their own school progress

2.1.2. Teacher

Teachers can use the system to keep the details regarding class, student progress and classroom activities in one place. It’s easy to view a particular teacher’s class schedule.

2.1.3. School System Administrator

System Administrator will use the system to login, create, modify, and deleted systems users.

2.1.4. Parents

Parents will be able to view their kids' school progress report which will help them make calculated decision towards their kids' schools' related issues

3. PROJECT CHARACTERISTICS, REQUIREMENTS AND USER ACCEPTANCE CRITERIA

3.1.1. CHARACTERISTICS

- Open source project
- Target audiences are students and educators
- Web based system
- Easily customizable

3.1.2. REQUIREMENTS

- Workstation or PC for each team member
- Code Editor (Visual Studio Code, Brackets, Notepad++)
- GitHub Version Control
- Project installed and running on each PC

3.1.3. USER ACCEPTANCE CRITERIA

- Users:
    - Students: Successfully login and view end of semester report
    - Teachers: Successfully login and view semester report for their subject(s)
- Acceptance on the Git and Github repository:
    - Individual and teamwork contributions committed to the cloud repository that help complement each group members work done as we strive to achieve one goal.
- Sign off by supervisor lecturer

4. ROLES AND RESPONSIBILITIES

Name	Roles and Responsibilities
Kephas Shiweda	Project Manager / Database Administrator
Erich Godenschweig	Documentation / User Interface Programming
Fikameni Petrus	Tester / Backend Database Programming
Dan Ntwari	UI Designer / Programming

5. TASK ASSIGNMENT, COLLABORATION AND DEVELOPMENT TOOLS

During the implementation of this project our team will use tools such as the ones mentioned in Table 1 - Tools.

Name of Tool	**Purpose**	**Benefit**
Trello	Web-based list-making application for task assignments.	Improved collaboration and task assignment.
Google Drive	Documentation and file sharing tool.	Improved collaboration and documentation control.
Brackets	Integrated Development Environment	Programming and compilation tool to produce the necessary software additions.

Table 1 - Tools

6. PROJECT DELIVERABLES

Each phase of the project will include core outputs and key deliverables. These are outlined in Table 2 - Deliverables and Outputs.

	Deliverable	Output
Progress 1	Project plan	Documentation report on what is to be done and expected from project group members. Report will outline Project overview, addition of functionality and roles and responsibilities.
Progress 2	Development progress and status. Mock-ups, wireframes and user interface diagrams are delivered during progress 2.	Teamwork as well individual capabilities and contribution thus far. Display progress on testable software. Diagramming and progress explanations.
Progress 3	Demo and documentation of new functionality.	Final showing of the new functionality and the documentation of work done. New functionality should be tested and test scripts supplied.

Table 2 - Deliverables and Outputs

7. ADDING FUNCTIONALITY

- Generate report for:
  - Student's final assessments grade on each subject enrolled
  - End of the semester report for each student. Viewable online on their dashboard.

- Teacher's query result:
  - Data mining tool for teachers to query average, maximum, minimum grades, etc.

8. WORK BREAKDOWN STRUCTURE

Deliverables / Milestones	Dates	Responsible Person(s)	Revised date & reason

1. Start with Pre-initiation: NUST – Supervised Team meeting.	11 February 2019	Erich Godenschweig, Dan C. Ntwari, Kephas Shiweda , Fikameni Petrus, Mr Colin Stanley.	
2. Kick off Meeting with Project Team	24 February	Erich Godenschweig, Dan C. Ntwari, Kephas Shiweda , Fikameni Petrus.	
3. Present project plan to the professor.	13 March	Erich Godenschweig, Dan C. Ntwari, Kephas Shiweda , Fikameni Petrus, Mr Colin Stanley.	
4. Functionality design.	13-16 March	Erich Godenschweig, Dan C. Ntwari, Kephas Shiweda , Fikameni Petrus	
5. Speak to professor on progress. Provide wireframes and mock-ups.	16 March	Erich Godenschweig, Dan C. Ntwari, Kephas Shiweda , Fikameni Petrus, Mr Colin Stanley.	
6. Final functionality coding and documentation.	13-17 March	Erich Godenschweig, Dan C. Ntwari, Kephas Shiweda , Fikameni Petrus	
7. User Acceptance Testing (UAT).	18 March	Erich Godenschweig, Dan C. Ntwari, Kephas Shiweda , Fikameni Petrus	
8. Sign-off Project Charter by Professor and Team.	13 March	Mr Colin Stanley	

9. ACTIVITY LIST

Activity Identification or Number	Activity Name	Activity Description	Activity Attributes	Predecessors	Milestone?

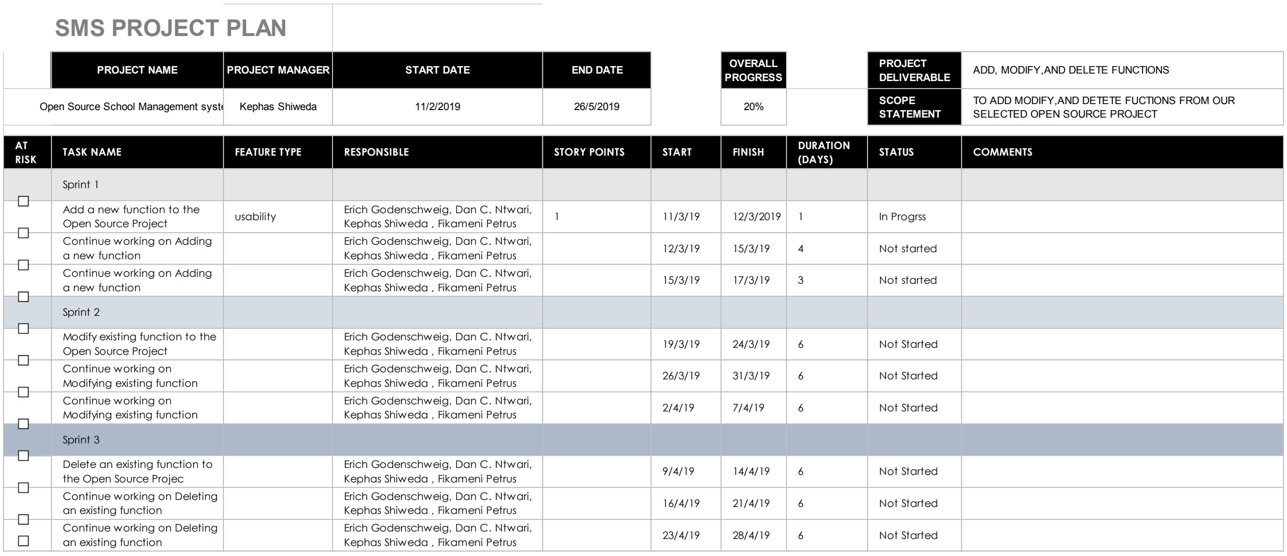
1.	Start with Pre-initiation: NUST – Supervised Team meeting.	A pre-meeting with the NUST team members and the professor.			Yes
2.	Kick-off Meeting with Project Team	A meeting with the whole team			Yes
3.	Gather Customer Requirements via Site Visit.				
4.	Implement Addition of Software Functionality.	Implement the messaging functionality, to notify people when a sales order has been done on the ERP system			
5.	Functionality Documentation.	Document the newly added functionality for future reference			
6.	Implement Removal of Software Functionality.	Fulfill removal of functionality			
7.	Functionality Documentation.	Document and note the deletion functionality for future reference			
8.	Implement Modification of Software Functionality.	Fulfil modification of functionality			
9.	Functionality Documentation.	Document the modified functionality for future reference			

10.	User Acceptance Testing.	Demo the software for the user/professor			
11.	Sign-off Project Charter by Professor and Team.	Final sign-off			

Our project follows an agile project plan which is based on features. The plan estimates how long it will take for each feature to be delivered, without much detail on how it will be delivered. And because the project plan is focused on features, we group similar features into sprints.

Once the plan is developed, the project team needs to maintain it and update status and timelines accordingly.

Also known as an agile project schedule, this plan lets you add your tasks, who is responsible, start and end dates, and status. The duration for each task will be automatically calculated. Below is a screen shot of our project plan



SMS PROJECT PLAN

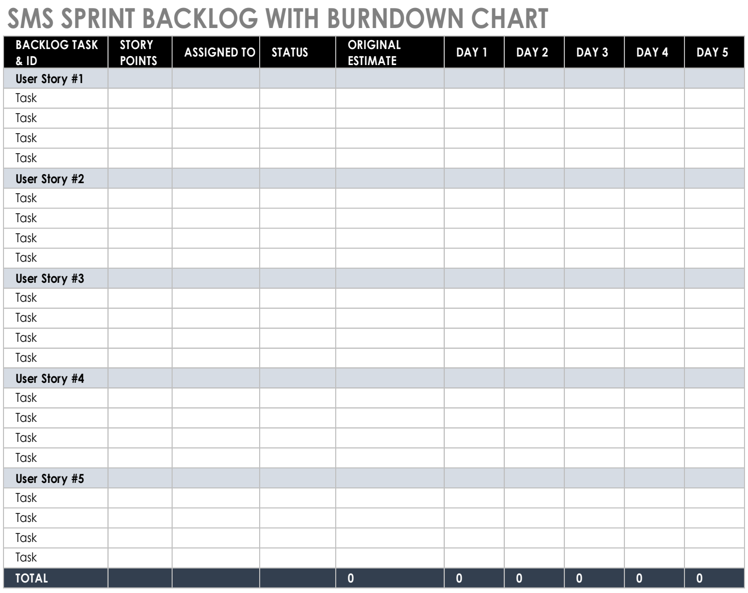
PROJECT NAME	PROJECT MANAGER	START DATE	END DATE	OVERALL PROGRESS
Open Source School Management syste	Kephas Shiweda	11/2/2019	26/5/2019	20%

PROJECT DELIVERABLE	ADD, MODIFY,AND DELETE FUNCTIONS
SCOPE STATEMENT	TO ADD MODIFY,AND DETETE FUCTIONS FROM OUR SELECTED OPEN SOURCE PROJECT

AT RISK	TASK NAME	FEATURE TYPE	RESPONSIBLE	STORY POINTS	START	FINISH	DURATION (DAYS)	STATUS	COMMENTS
□	Sprint 1								
□	Add a new function to the Open Source Project	usability	Erich Godenschweig, Dan C. Ntwari, Kephas Shiweda , Fikameni Petrus	1	11/3/19	12/3/2019	1	In Progrss	
□	Continue working on Adding a new function		Erich Godenschweig, Dan C. Ntwari, Kephas Shiweda , Fikameni Petrus		12/3/19	15/3/19	4	Not started	
□	Continue working on Adding a new function		Erich Godenschweig, Dan C. Ntwari, Kephas Shiweda , Fikameni Petrus		15/3/19	17/3/19	3	Not started	
□	Sprint 2								
□	Modify existing function to the Open Source Project		Erich Godenschweig, Dan C. Ntwari, Kephas Shiweda , Fikameni Petrus		19/3/19	24/3/19	6	Not Started	
□	Continue working on Modifying existing function		Erich Godenschweig, Dan C. Ntwari, Kephas Shiweda , Fikameni Petrus		26/3/19	31/3/19	6	Not Started	
□	Continue working on Modifying existing function		Erich Godenschweig, Dan C. Ntwari, Kephas Shiweda , Fikameni Petrus		2/4/19	7/4/19	6	Not Started	
□	Sprint 3								
□	Delete an existing function to the Open Source Projec		Erich Godenschweig, Dan C. Ntwari, Kephas Shiweda , Fikameni Petrus		9/4/19	14/4/19	6	Not Started	
□	Continue working on Deleting an existing function		Erich Godenschweig, Dan C. Ntwari, Kephas Shiweda , Fikameni Petrus		16/4/19	21/4/19	6	Not Started	
□	Continue working on Deleting an existing function		Erich Godenschweig, Dan C. Ntwari, Kephas Shiweda , Fikameni Petrus		23/4/19	28/4/19	6	Not Started	

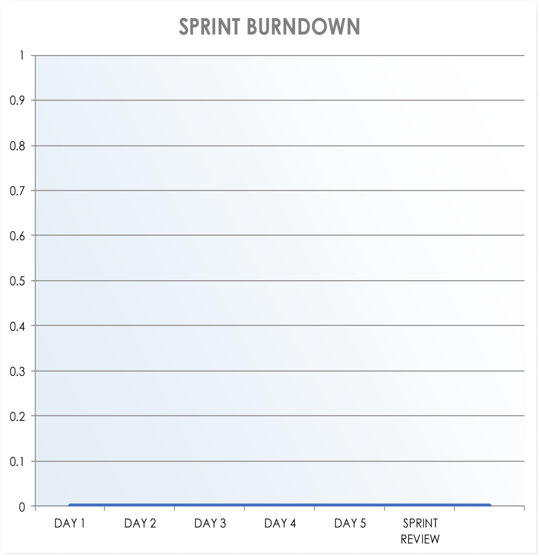
Also known as a scrum backlog, the sprint backlog is created during agile sprint planning, where the team will select the top items in the product backlog and add them to their sprints. The sprint backlog includes all the work pushed into the development phase. It is a to-do list of backlog items that must be completed in the current iteration and this list should be very finalized (no one should be adding or removing tasks).

The backlog has columns for backlog item, story points, responsible, status, and original estimate. In the day 1 through day 5 columns, you can add the number of extra development hours required each day for the task. In the *Total* row at the bottom, you'll see the total amount of extra development hours for each day for all tasks in the sprint. The burndown chart then represents this outstanding work. Below is our Sprint backlog with a burndown chart.



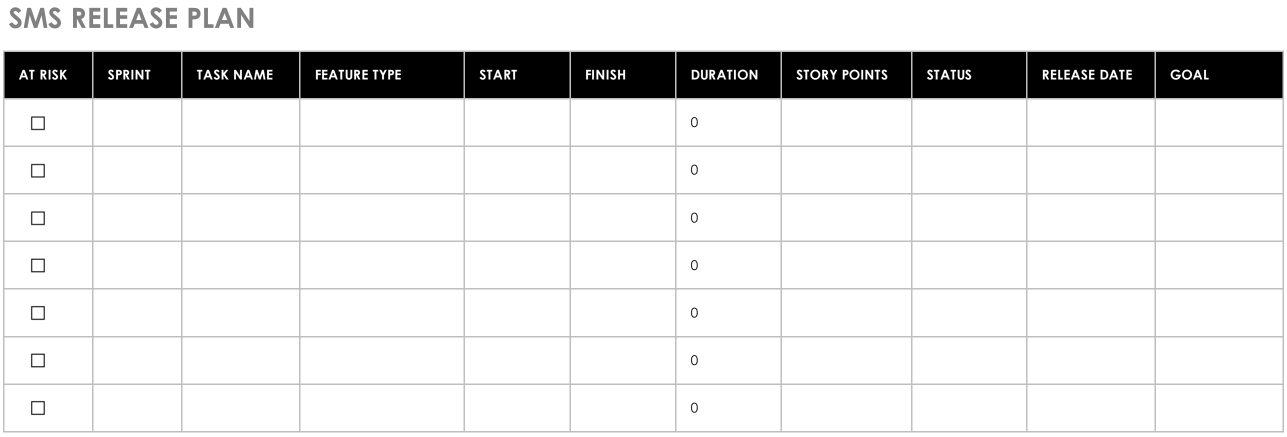
SMS SPRINT BACKLOG WITH BURNDOWN CHART

BACKLOG TASK & ID	STORY POINTS	ASSIGNED TO	STATUS	ORIGINAL ESTIMATE	DAY 1	DAY 2	DAY 3	DAY 4	DAY 5
User Story #1									
Task									
Task									
Task									
Task									
User Story #2									
Task									
Task									
Task									
Task									
User Story #3									
Task									
Task									
Task									
Task									
User Story #4									
Task									
Task									
Task									
Task									
User Story #5									
Task									
Task									
Task									
Task									
TOTAL				0	0	0	0	0	0



Agile release planning happens during sprint zero, when there is no product to deliver and the team can instead focus on defining the release goal, the features that need to be delivered, assigning features to a sprint, and estimating the duration of each task. Release planning may change as new stories are added or deleted.

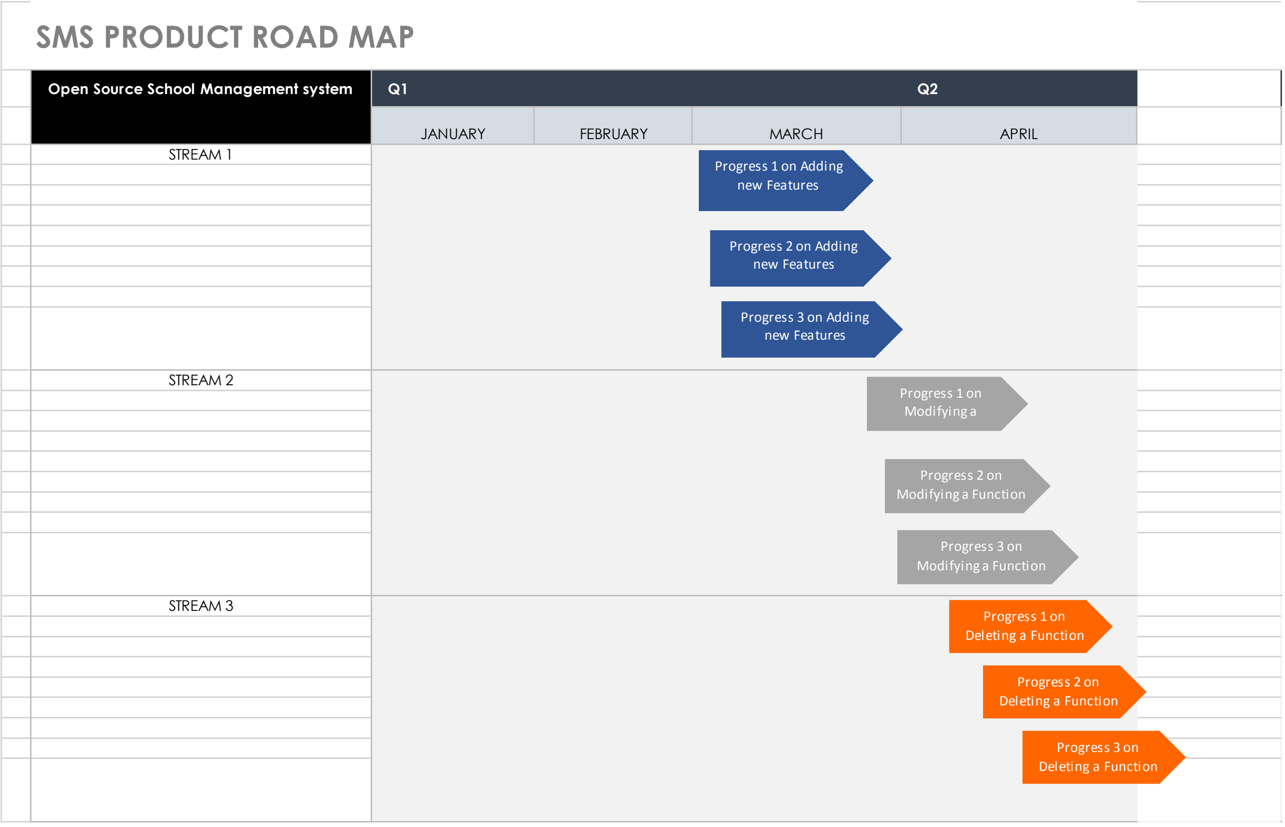
This agile release plan allows you to list all your tasks, assign each task to a sprint, and calculate the duration based on start and end dates. You can also indicate the status of each task from the drop-down menu and define each corresponding goal. Below is our Release plan.



SMS RELEASE PLAN

AT RISK	SPRINT	TASK NAME	FEATURE TYPE	START	FINISH	DURATION	STORY POINTS	STATUS	RELEASE DATE	GOAL
□						0				
□						0				
□						0				
□						0				
□						0				
□						0				
□						0				

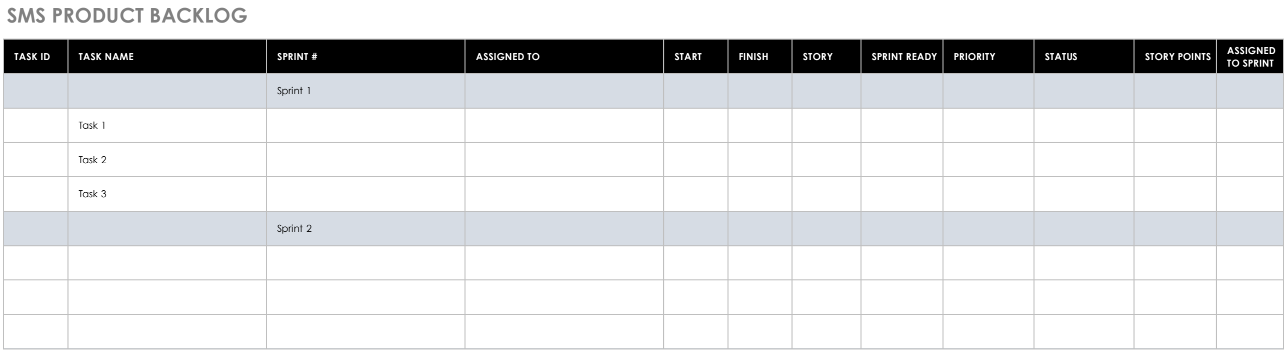
An agile roadmap represents a strategic overview of where the product is headed in the mid-to long-term. It steers the direction of your product and sets expectations within your project. A traditional roadmap can sometimes act as a strict project plan, but in an agile project, the roadmap just provides guidance and clarity. Below is our Agile road map strategic overview.



A product backlog helps the product owner keep track of all the features that stakeholders would like to include in the product. In other words, the product backlog is kind of like a wish list of all possible features in the final product.

Unlike a sprint backlog, the product backlog is always changing and improving. Anyone can add features to the product backlog, with the product owner prioritizing each one.

This agile product backlog plan includes drop-down columns for story, sprint ready, priority, status, story points, and assigned to sprint. It also automatically calculates the total story points for each sprint based on the points for each task. Below is our product backlog plan.

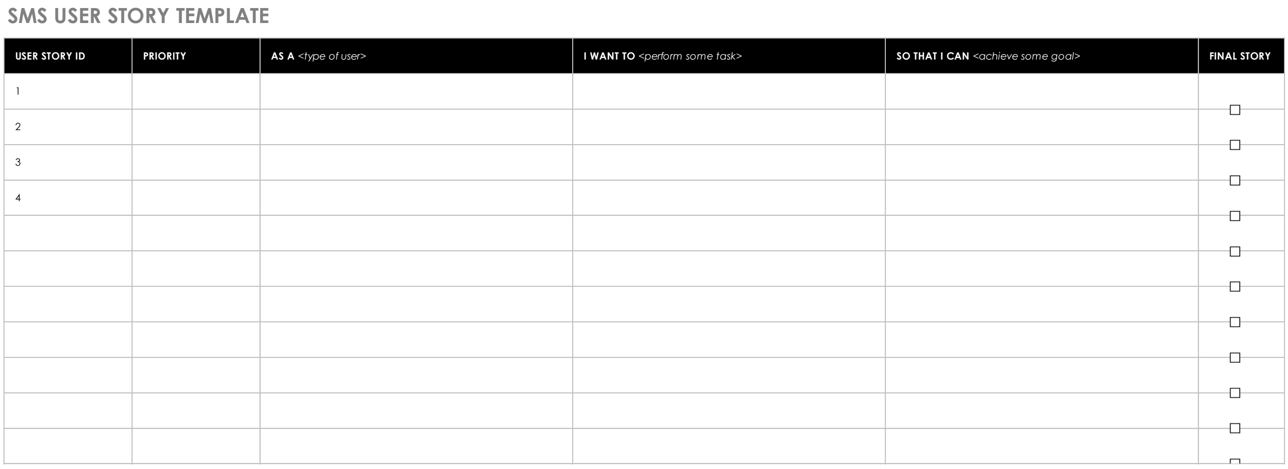


SMS PRODUCT BACKLOG

TASK ID	TASK NAME	SPRINT #	ASSIGNED TO	START	FINISH	STORY	SPRINT READY	PRIORITY	STATUS	STORY POINTS	ASSIGNED TO SPRINT
		Sprint 1									
	Task 1										
	Task 2										
	Task 3										
		Sprint 2									

A user story describes a feature from the end-user’s perspective. It includes the type of user, what they want, and why they want it. These short, one-sentence user stories create a super simple description of a requirement. Then, the development team develops code that will satisfy the requirements of the user story.

This agile use case plan follows the typical agile story structure: as a <type of user>, I want to <perform some task> so that I can <achieve some goal>.Below is screen shot of our user story.

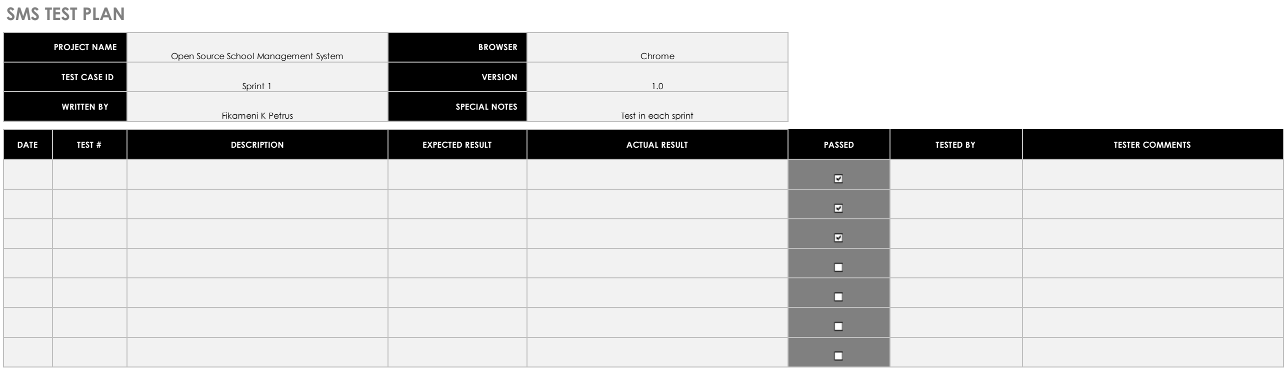


SMS USER STORY TEMPLATE

USER STORY ID	PRIORITY	AS A *<type of user>*	I WANT TO *<perform some task>*	SO THAT I CAN *<achieve some goal>*	FINAL STORY
1					☐
2					☐
3					☐
4					☐
					☐
					☐
					☐
					☐
					☐
					☐
					☐

Instead of a static testing plan that must happen at a certain time, test plans in agile projects should be dynamic and iterative. The testing phase becomes an extension of the requirements prioritization process so that the most up-to-date information is used when defining tests and to avoid any misunderstanding about scope.

While you don't need an extensive agile test plan, you still need to track the actions, expected results, actual results, and whether the test passed or failed. This plan includes interactive checkboxes in the *Pass* column, so when you check the box that the test passed, the whole row will be crossed out. Below is a screen shot of our test plan.



SMS TEST PLAN

PROJECT NAME	Open Source School Management System	BROWSER	Chrome
TEST CASE ID	Sprint 1	VERSION	1.0
WRITTEN BY	Fikameni K Petrus	SPECIAL NOTES	Test in each sprint

DATE	TEST #	DESCRIPTION	EXPECTED RESULT	ACTUAL RESULT	PASSED	TESTED BY	TESTER COMMENTS
					☑		
					☑		
					☑		
					☐		
					☐		
					☐		
					☐		

Annexure: ASD Gantt Chart.