Software Requirements Specification

for

Resala School Suez Website

Version 1.0 pending approval

Prepared by Mohmed Ashamallah

Resala School Suez

Jan-2021

Table of Contents

Revision History

Name	Date	Reason for changes	Version
Mohamed Ashamallah & Yasmin Ashraf	17-1-2021	Create chapter 1.	0.1
Mohamed Ashamallah	18-12-2021	Finish SRS without any function details or diagrams or UI or wireframe design	0.2

1. Introduction

1.1 Purpose

The main purpose of the **website** is to enhance identify **Resala School Charity organization**, help the **organization** to get donations, and help the **organizers in organizing all organization's** events.

The purpose of the SRS document to help all front-end developers, back-end developers, designers, UX designers, and the **client team** to understand all features on the website and what we expect.

Make sure is every member of the team and **organization** knows how we work, the plane of work, and all tools we use for every part (design, development, management).

Help the future developers to understand the **system** under the hood to perform better in development.

Make sure the volunteer developers know **what they are against** and discover if they can deal with it and can add value to the project and themselves or not, to save everyone time.

1.2 Document Conventions

- The green color used for reference to the project itself.
- The orange for system admins, the organization, and its member.
- The blue for referring in the glossary.
- The red for References.
- For a tool use purple color

1.3 Intended Audience and Reading Suggestions

- Organization members (Marketing, Media, ...etc.).
- System developers (front-end developers, back-end developers, designers, UX designers).

- Volunteer developers.
- Future developers.

1.4 Product Scope

Helping different types of users such: **organization admins**, teachers, students, volunteers, and interested people.

By organizing events, courses, travels, blog posts, and parties for volunteers and interested people.

Also, this can help the **organization** to collect donations, selling their own products, and helping the students to get their study material and the academic schedule.

It will be more specific in the next chapters.

1.5 References

Egypt top web browser for 2020.

https://gs.statcounter.com/browser-market-share/desktop/egypt

https://gs.statcounter.com/browser-market-share/tablet/egypt

https://gs.statcounter.com/browser-market-share/mobile/egypt

Charity example.

https://themeforest.net/search/charity

Tools

https://moqups.com/

https://app.diagrams.net/

Laravel security layers

https://auth0.com/blog/why-laravel-is-the-recommended-framework-for-secure-mission-critical-applications/

https://www.cloudways.com/blog/best-laravel-security-practices/

2. Overall Description

2.1 Product Perspective

The **website** will be a standalone system that helps the organization, there is no tech bass or API to work with, but there is a third-party API the system will deal with like pay API.

The system not to change the old way but for a better way.

2.2 Product Functions

We must care about many different types of users.

The normal user is a user who wants to read content from the blog or buy something from the site or he just wants to pay donations, this type of user I very important because I get revenue one way or another, it must be so simple and easy experiences for him.

The second function is that care about education, so we have two main users one is a teacher, and he can be old and hate using technology so I need to make it very much easy use and very specific and quack, and student **I must** care he can be naughty like saying bad words in chat, so **we must** make a control and block by **admins** to his own safety.

The last main function is the one cares about adding content to the **system** and it will be by **admins**.

2.3 User Classes and Characteristics

Super Admin: This type of user can give the other admin their powers and he can **CRUD** any data on the website like user data, admin data, products, posts.... etc.

Admin: one of the **origination members** the super admin gives him his power to do something like **CRUD** posts, products, users .... etc.

or just see the messages from contacts and answer them.

we work in the **SRS** like he has all powers he can have.

He cannot create his own account.

Teacher: this user has very little power like he can add study material or chat with the students or other teachers. also, he can add exams for students and see the result to give his review.

He cannot create his own account.

Student: this user can take exams or material study.

Also, he can see the academic schedule or just chat with his teacher.

He cannot create his own account.

Normal use: This type of user can buy a product or can **CRUD** his post or read like, comment, or share any other posts in the blog.

He can create his own account.

Website viewer: he is not a member or has an account on the site he just wants to see the different types of events or see the gallery or see the product or maybe see the blog.

He cannot add any of this stuff, but he can share on social media or share links.

2.4 Operating Environment

The system is a website, so it is a web-based application. as we now ever website have to main faces front face and back-face. so, the web site will work in web browsers, but the data and the back end will work in a server or a host.

2.5 Design and Implementation Constraints

For UI you must live with the color of the logo, and I hope if you look at examples in the **reference section**.

The user experience must be as smooth as it is possible.

For the wireframe, I hope if we just can look at **moqups** in the **reference section – tools**, no wireframe program is necessary.

Not all screens have the wireframe so for the final UX designs we will use **Adobe XD**.

The front-end must be sure of every feature work excellent in **Google chrome**.

some of the features must update in real-time like a chat system or at least update every few seconds it will work with **AJAX**.

The main front-end tools are (**HTML5, CSS3, ES6, Bootstrap 5, SCSS, NPM, Webpack, ESLint**).

For the back end, we use **Laravel** and **MySQL**.

For project Version control using **Git** and **GitHub** as a git host.

For project plan management we use **Trello**.

For analyzing design, we use **Diagrams** in the **reference section – tools**.

2.6 User Documentation

We will make the **system** UX easy as it is possible. but if we need to create user documentation it will be an FQA for every type of user. viewer and user FQA, Super Admin documentation, student FQA, teacher FQA, and the tricky one is admin FQA because it will be different with different admin power.

2.7 Assumptions and Dependencies

The organization has so many management departments, the developer can care about every single department and every admin need and want, so the admin power must be very flexible.

A lot of our users and viewer do not have time to see every single part of the website, so I must make sure they can get what they want as easily as possible.

Because of the very low budget, the server will be weak, can take ~100 000 visits monthly.

3. External Interface Requirements

3.1 User Interfaces

For UI you must live with the color of the logo, and I hope if you look at examples in the **reference section**.

The user experience must be as smooth as it is possible.

data area in a multitasking operating system), specify this as an implementation constraint.>

3.2 Communications Interfaces

For the budget issue, we will get the best features we can get from the Shared host. so we don't have much choice.

The system has a notification and sends an email.

Some of the messages will be in spam because of the shared host.

4. System Features

This Section is under construction.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

The system will be able to work on Tablet and mobile (Responsive Design).

We use web-bundle like webpack for optimizing and minimize the code as it is possible.

5.2 Safety Requirements

If the web host does not give a backup plan, The **super Admin** Must Backup the web data weekly.

5.3 Security Requirements

For the front-end, we must have client-side validation.

For the back end, we use Laravel to give us good security layers.

We must be sure the host is safe and give us a layer of safety.

Use a good hash system.

If there is a software tester it will be much better.

5.4 Software Quality Attributes

You must be organized in view (front-end) part as you can.

Use Git workflow in development mode.

5.5 Business Rules

The Super Admin and Admins must have good knowledge of using computer and internet because of his power.

Appendix A: Glossary

CRUD: Create, Read, Update, Delete item form Database.

Appendix B: Analysis Models

This Section is under construction.

Appendix C: To Be Determined List

This Section is under construction.