SYSTEMS IMPLEMENTATION PLAN

PROJECT TITLE	AHPCZ MANAGEMENT INFORMATION SYSTEM

PROJECT OWNER	PROJECT MANAGER	PROJECT DEVELOPER	DATE INITIATED
AHPCZ	ZHD CONSULTING	LEADING DIGITAL	

PLANS	WHERE TO FIND
Description	This Document
Modules	This Document
Technologies	This Document
Testing Strategy	This Document
Deployment	This Document
Environmental Control Approach and Maintenance	This Document
Knowledge Transfer and Training Plan	This Document
Operational Impact Specification	This Document

SYSTEM MODULES / COMPONENTS

MODULES

MODULE NAME	DESCRIPTION OF FUNCTION	SOURCE
Practitioner details	Add, edit and update personal information	Portal or documents
Practitioner contacts	Add, edit and update contact information	
Practitioner qualifications	Add, edit and update qualification information	
Practitioner Employment	Add, edit and update employment information	
Practitioner payments	Record and verify all payments	
Practitioner checklist	Verify required information and documentation	
Practitioner approval process	Step by step checks and balance of submitted information for application approval	
Practitioner Practicing Certificates	Generate required certificates	
Practitioner renewals, penalties and restoration	Calculate all renewal fees, penalties and restorations fees on payment	
Portal Integration	Provide access to the system from practitioners' portal	

TECHNOLOGIES USED

TECHNOLGY PIECE	TECHNICAL DESCRIPTION	USAGE DESCRIPTION
Backend Development	PHP 7.4, Laravel Framework version 8.0	<Purpose, frequency of updates, stability, etc.>
Front End Design	HTML 5, CSS, Bootstrap framework 4.2 and JavaScript	System front design
API Integration	Guzzle HTTP 3.0	For all API calls
Dynamic Components	Laravel Livewire version 2.0	Reactivity on Practitioner and Student Lists tables
Database	MySQL 5	SQL queries

DEVELOPMENT APPROACH

This system was developed using the Agile approach and as such all components have Models that connect to each other which allows an additional module to be added without any difficulties.

SYSTEM ARCHITECTURE - Model View Controller (MVC)

Architecture	Location folder	USAGE DESCRIPTION
Models	App/Model	Make database and relationship calls
Views	Resources/views	renders all the content, this is the front-end interface.
Controllers	App/Http/Controllers	Makes calls between models and Views, data manipulations, calculation and passes processed data to the view
System Routes	Routes/web	Stores All URLs used to access the system
API routes	Routes/api	Stores All API calls URLs
Migrations	Database/migrations	Store all database tables and structures

INTEGRATION APPROACH

This section identifies dependencies and the sequences in which components will be integrated. All dependencies are installed using Composer a tool that is already installed within the system.

Dependencies	Package	USAGE DESCRIPTION
Laravel Livewire	"livewire/livewire": "^2.3",	Reactivity on practitioner and student lists
DOMPdf	"barryvdh/laravel-dompdf": "0.8.7",	Generating Certificate and ID cards
Excel Management	"maatwebsite/excel": "^3.1",	Import and export excel documents
Simple QR Code generator	"simplesoftwareio/simple-qrcode": "^4.2"	Generate QR Codes
Guzzle HTTP	"guzzlehttp/guzzle": "^7.2",	Make All API calls
Paynow	"paynow/php-sdk": "^1.0",	Online Payments

DEPLOYMENT ENVIRONMENT

REQUIREMENT	TECHNICAL DESCRIPTION	REQUIREMENTS
SERVER env	LINUX/WINDOWS	PHP 7.3 installed windows or mac
Database	MYSQL	Version 5 and above
Cpanel	Htaccess files/web config	Show hidden files
Production	IP address/domain name	
<Other Environments>		

TESTING STRATEGY

PLANNED TESTING ACTIVITIES

TYPE OF TEST	DESCRIPTION	TEST	FREQUENCY
Unit Testing	Test each module using a resource controller, model and database table for specific test	DONE	Every change and update
Integration Testing	Integrate modules, create model relationships test using Tinker (php artisan tinker) in your terminal	DONE	Every change x minutes
Load Testing	Test speeds over https using postman	DONE	On every request
User Acceptance Testing	User experience, speeds, functionality and output.	Done	All modules on deployement
<Other Tests>			

ENVIRONMENT CONTROL APPROACH AND MAINTANENCE

Constantly updating the Php version and MySQL on the server for a period of every 6 months will help us maintain the latest security patches. Updating of All dependencies should be done regularly every 2 months.

KNOWLEDGE TRANSFER AND TRAINING PLAN

KNOWLEDGE REQUIREMENTS

KNOWLDEGE AREA	KNOWLEDGE GROUP	REQUIRED OPERATIONAL GROUPS	
technical Knowledge	ZHD Developers	Developers and Hosting Manager	
System Knowledge	AHPCZ Staff	All departments	
Application Knowledge	Developers	Developers and technical Support	

KNOWLEDGE TRANSFER PLAN

TRANSFER ACTIVITIY	AUDIENCE	PERSON RESPONSIBLE	TIMEFRAME
User Manual	AHPCZ Staffa	All departments	6 hours
Developer Workshop	ZHD Staff	Developers	2 Weeks

TRAINING REQUIREMENTS

USER GROUP	TRAINING NEEDS	SIZE OF GROUP	LOCATION OF GROUP
<Admins>	<Roles, modules, functions>	<# people>	<HQ, communities>
<Users>			
<Other Groups>			

OPERATIONAL IMPACT SPECIFICATION

OPERATIONAL PROFILE

Operating Hours	<7 days - 24 hours per day, 6 days - 22 hours per day>
Expected Availability	<high availability: 99.5%>
Expected Reliability	<fault tolerance: 99.9%>

Peak Busy Hours	<09:30 - 10:30 hours, 13:00 - 14:00 hours>
Maximum Tolerable Outage	< 4 hours during maintenance and support>
Backup Window	<daily between 23:00 - 24:00 hours, weekend availability>
Backup Requirements	< full backup once a week, off-site requirement>
<Other Specifications>	