Laze

Table of Contents

1. **Introduction**
    1. Purpose
    2. Scope
    3. References
    4. Overview
2. **Interface Description**
    1. Module Interface
3. **Detailed Design**
    1. Module Detailed Design
    2. Module Decomposition
    3. Coding standard & Documentation
4. **Versions**

Introduction

Purpose

This software design document states how the requirements will be achieved by describing the architecture and system design of the Laze web application. This document will provide descriptions of how each module will work including their associated methods and attributes.

Product Scope

The scope of this project is to continuously provide live updating information for users around Wilfrid Laurier campus. Users will have the ability to set up their own data points as well as gain information from other user data points around them.

References

IEEE. IEEE Std 1016-1998 IEEE Recommended Practice for Software Design Descriptions.
IEEE Computer Society, 1998
CS312 - Software Engineering. Software Design Document for Electronic Scrolling Displays.
TGR Karthik, 2006

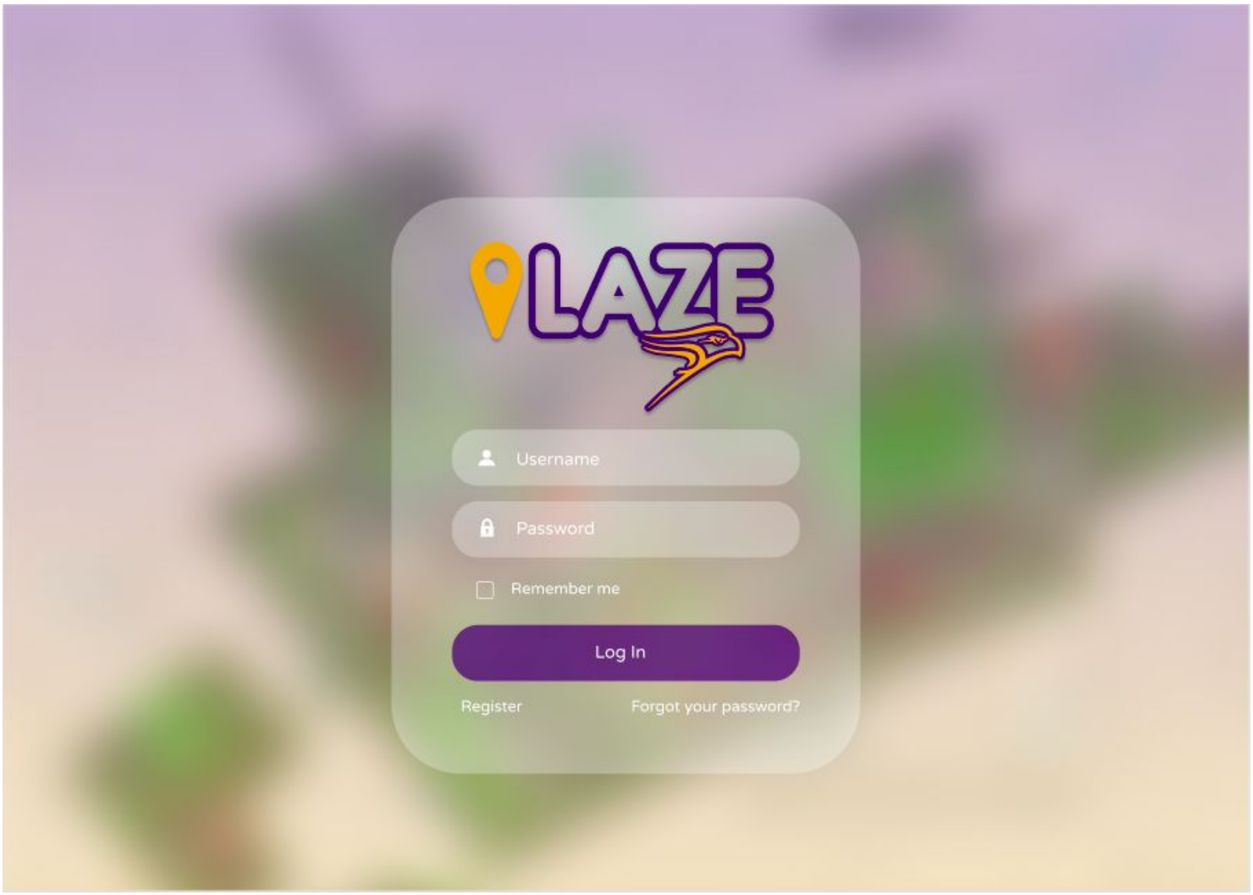
Overview

This document is written following the guidelines of the IEEE Recommended Practice for Software Design Descriptions. It outlines the module, process, and data dependencies, as well as their detailed design and decomposition.

2. Interface Description

2.1 Module Interface

2.1.1 LoginView



Field	Type	Description
forgot-password	button	Creates dialogue with a textbox to prompt user to enter recovery email address.
guest-login	button	Allows the user to login as guest to view the MainView.
login	button	Submits user credentials and logs them into MainView if they are correct.
password	input text	The user’s login password.

register	button	Opens dialogue box that requests signup info.
rememberMe	button	Radio button user can select to save username.
username	input text	The user’s login name.

2.1.1.1 RegistrationView

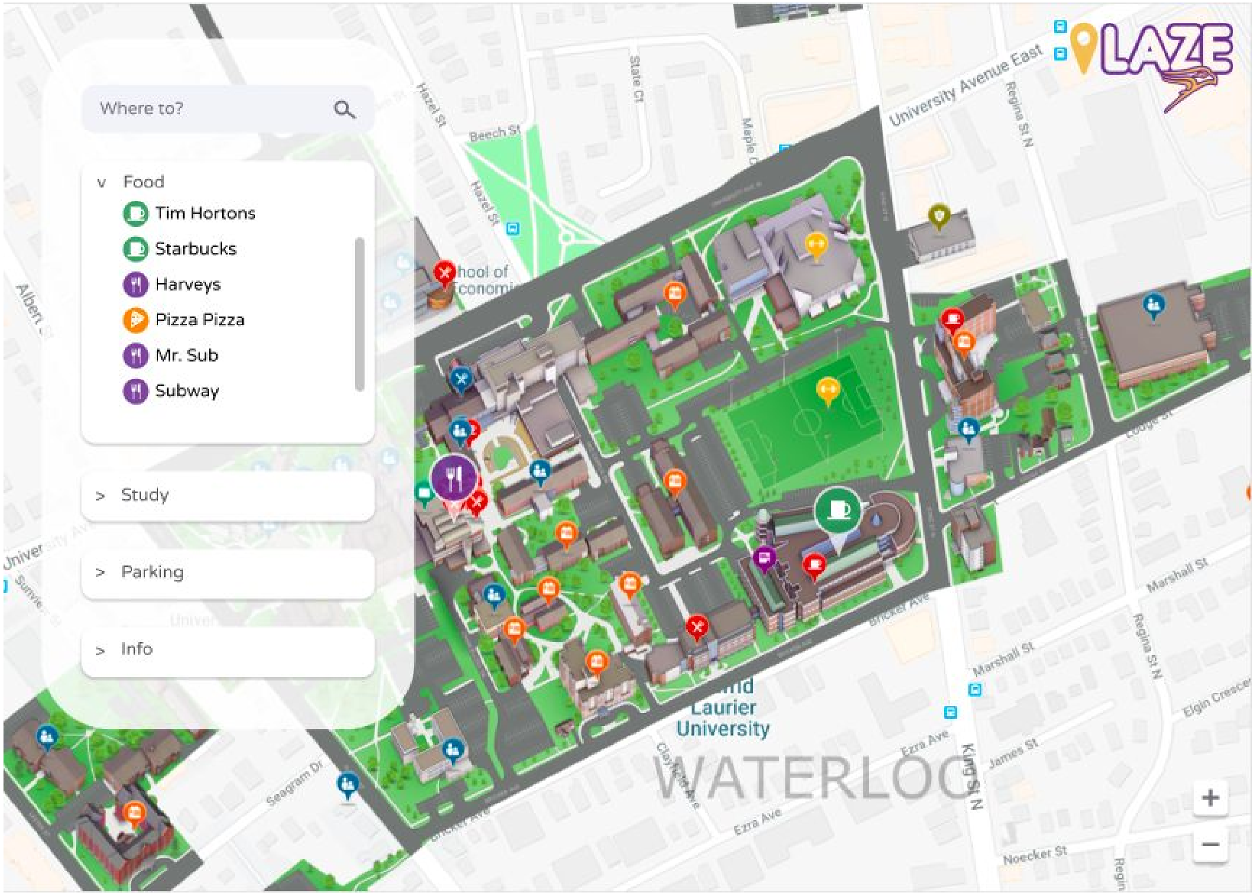
Field	**Type**	**Description**
confirm-password	input text	User confirms their desired password.
email	Input text	The user’s confirmation email address.
password	input text	The user’s desired login password.
password-check	label	Dynamic label that displays any errors regarding passwords not meeting requirements or if the two passwords don’t match.
register	button	Registers the account and sends out confirmation email.
terms-of-service	button	The text will be a link that if clicked will send the user to the terms of service.
tos-agreement	radio button	User checks this if they agree to the terms of service.
username	input text	The user’s desired name.
username-availability	label	Dynamic label that displays if the entered username is available.

2.1.1.2 ForgotPasswordView

Field	**Type**	**Description**
confirmation	label	Dynamic label that tells the user if the email was sent successfully or if the email entered is not in the database.

email	input text	User enters the recovery email address registered when the account was created.
send	button	Submits the email entered to receive the password-change email.

2.1.2 MainView



Field	Type	Description
addPin	button	Brings up the CreatePinView.
food	drop down	Drop down menu that displays all current food related pins.
study	drop down	Drop down menu that displays all current study related pins.
parking	drop down	Drop down menu that displays all current

		parking related pins.
info	drop down	Drop down menu that displays all current info related pins.
mapPin	button	Any active pin on the map when clicked will bring up a dialogue containing the details of that pin.
whereTo	input text	Enter keywords to search pins.
zoomIn	button	Zooms the map in.
zoomOut	button	Zooms the map out.

2.1.2.1 CreatePinView

Field	**Type**	**Description**
category	drop down	Drop down menu that allows the user to select the category of their pin (ex. Food, study, parking, etc.).
details	input text	User enters the details of their pin.
exit	button	X button that allows the user to cancel pin creation and return back to MainView.
mapselector	map	User clicks on the map where they would like to place their pin.
submit	button	Submits the user’s pin to be posted.

3. Detailed Design & Decomposition

3.1 Module Detailed Design

3.1.1 Login View, RegistrationView, & ForgotPasswordView

Function Name	**Parameters**	**Description**
CreateUser()	Email address, name, username, encrypted password, profile photo (optional)	Creates a user in the database. Returns the result of action (i.e. it worked, or there was an error).
DeleteUser()	Email address, encrypted password	Deletes the user from the database. We must decide which user data will be orphaned and which data will be deleted.
Login()	Email address, encrypted password, rememberMe (boolean)	Takes login information and allows or denies the user access to the next page.
ResetPassword()	Email address	ResetPassword takes an email address, and if it matches an email address in the system, it sends a password reset link to that address.
SetUserData() (setters of the User class)	Email address, New user data	Functions relating to updating user data. The setters of the User class. Changes will be logged.

3.1.2 MainView (map page)

Name	Parameters	Description
DeletePin()	Email address, pin id	Will verify then remove the pin and related data. Action will be logged.
Search()	Query (string)	Searches the various aspects of pins for relevant information. Returns a list of pins.

3.1.3 Pins

3.1.3.1 CreatePinView

Name	Parameters	Description
CreatePin()	Email address, Pin location, pin type, pin description, pin name/title	Verifies information and creates a pin. Notifies the front end of the result. Logs information about pin creation.
SetPinData() (setters of the Pin class)	Email address	Pin data will be updated with setters of the Pin class. Changes will be logged.

3.1.3.2 Pin Class

Pin Class Properties

Modifiers	Type	Name	Description
private	int	ID	Uniquely identifies this pin.
private	String	Description	Describes the purpose of and reason for the pin.
private final	Point2D	Location	Describes the location of the pin on an XY plane (potentially we might

			want to replace this with a Point3D).
private final	EmailAddress	CreatedBy	The email address of the user who created this pin.
private	int	VoteCount	The sum of all upvotes and downvotes of this pin. This number should always be updated whenever a user votes or changes their vote. This number should only be changed by the UpVote() and DownVote() function, and should not have a direct setter function.
private	Collection<Vote>	VoterList	A collection of Vote objects which describes who has voted and how they voted.
private	String	Name	The name (short description) of the pin.
private	boolean	FlaggedAsBad	If someone has marked this pin as spam, this boolean should be set to true. Indicates that this pin might contain malicious information (like spam).
private	Collection<EmailAddress>	flaggers	A collection of the email addresses of the people who have flagged this post as bad.
private final	DateTime	dateTimeCreated	The date and time that this pin was created.
private final	DateTime	dateTimeToRemove	The date and time that this pin will expire. If NULL, this pin will never expire.

Pin Functions / Methods

Name	**Parameters**	**Description**
DownVotePin() Visibility: public	Email address Type: string	Decreases the Rating of [Pin ID], and records the

	Return type: int (rating of pin id)	rating and voter to VoterList of the pin.
FlagAsBad() Visibility: public	Email address, pin id, reason (enum), description Type: string,int,enum,string Return type: boolean	Allows users to flag pins as bad (spam, inappropriate, etc). Adds pin to a list of suggested actions for moderators to take.
Pin() Visibility: public	Name, Description, Location, CreatedBy Type: string,string,point2D,EmailAddress Return Type: ?	Constructor of the Pin object.
UpVotePin() Visibility: public	Email address Type: string Return Type: int	Increases the Rating of [Pin ID], and records the rating and voter to VoterList of the pin.

3.1.3.3 BusinessPin Class extends Pin

BusinessPin Properties

Modifiers	**Type**	**Name**	**Description**
private	BusinessHours	openHours	Describes the times when the business is open. Sets dateTimeToRemove to null by default.
private	int	congestion	Describes extent of how busy the business is.
private	Icon	businessIcon	A special picture that makes people think of this business.

3.1.3.4 Vote Class

Vote Properties

Modifiers	Type	Name	Description
private	EmailAddress	voter	Identifies the user who created this vote.
private	dateTime	createdAt	The date and time this vote was cast.
private	boolean	isUpvote	Describes the direction of the vote. True if the vote is an upvote, False if the vote is a downvote.

3.1.3.5 EmailAddress Class

Email Address Properties

Modifiers	Type	Name	Description
private final	String	Local-part	The part before the @ symbol in the domain name.
private final	String	domain	The domain the email is at. Should be wlu.ca or mylaurier.ca.
private final	dateTime	createdAt	The date and time this vote was cast.

Email Address Functions

Name	Parameters	Description
emailAddress	Local-part, domain	Constructor method which also verifies the information as described here (https://en.wikipedia.org/wiki/Email_address#Syntax). Will limit to a set

		of Domains if requested.

3.1.3.6 BusinessHours Class

BusinessHours Properties

Note: the setters for this class must verify that the opening time for each day is before the closing time.

Modifiers	Type	Name	Description
private	array<time>[7]	openingTimes	The time the business opens each day of the week. Day 0 is Sunday, Day 6 is Saturday.
private	array<time>[7]	closingTimes	The time the business closes each day of the week. Day 0 is Sunday, Day 6 is Saturday.

3.1.3.7 User

Modifiers	Type	Name	Description
private final	EmailAddress	emailAddress	The email address of this user. This address is used as a unique identifier for this user.
private	String	name	The real name of the user.
private	String	username	The displayed name of the user. Does not need to be unique.
private	EncryptedPassword	password	The password of the user. This property is

			sensitive, and needs to be handled carefully, and securely.
private	Image	profilePic	The picture the user wants to display on their profile

3.1.3.8 Moderator extends User

Being of the class Moderator gives the user some additional powers.

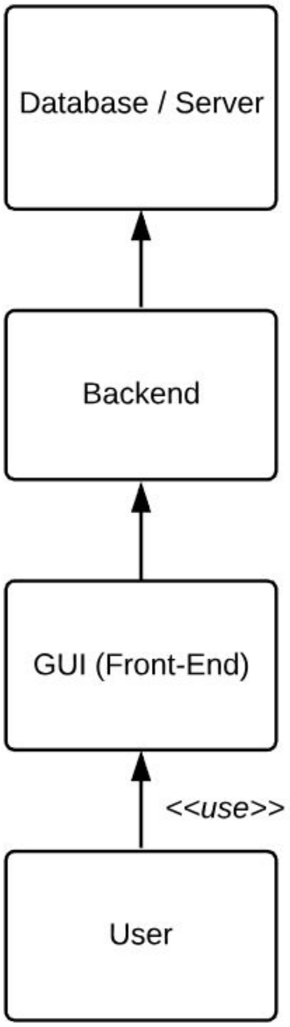
3.1.3.8 Administrator extends Moderator

Being of the class administrator gives the user some additional powers.

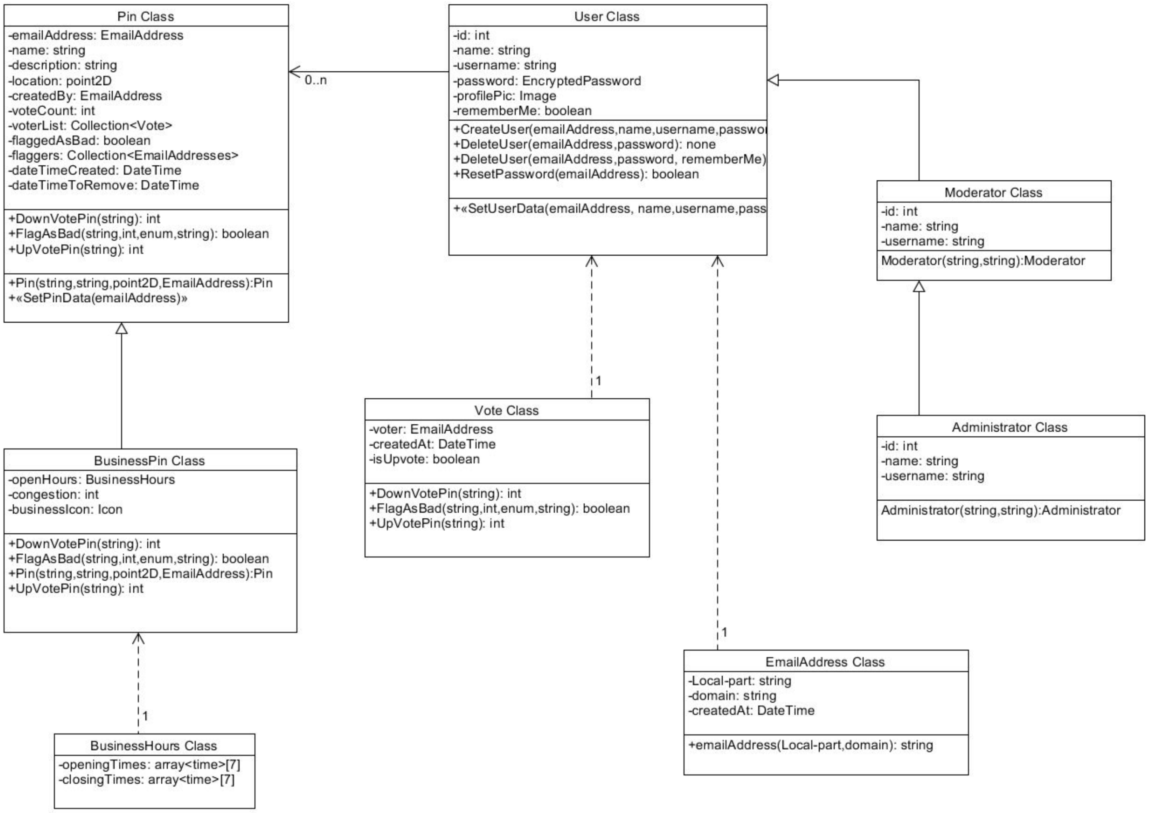
3.2 Module Decomposition

- Interaction diagrams
    - Framework Package Interactions
- Detailed class diagrams
    - Model Class Diagrams

Laze Framework Package Interactions



Model Class Diagram


Pin Class
-emailAddress: EmailAddress
-name: string
-description: string
-location: point2D
-createdBy: EmailAddress
-voteCount: int
-voterList: Collection<Vote>
-flaggedAsBad: boolean
-flaggers: Collection<EmailAddresses>
-dateTimeCreated: DateTime
-dateTimeToRemove: DateTime
+DownVotePin(string): int
+FlagAsBad(string,int,enum,string): boolean
+UpVotePin(string): int
+Pin(string,string,point2D,EmailAddress):Pin
+«SetPinData(emailAddress)»
0..n
User Class
-id: int
-name: string
-username: string
-password: EncryptedPassword
-profilePic: Image
-rememberMe: boolean
+CreateUser(emailAddress,name,username,passwo
+DeleteUser(emailAddress,password): none
+DeleteUser(emailAddress,password, rememberMe)
+ResetPassword(emailAddress): boolean
+«SetUserData(emailAddress, name,username,pass
Moderator Class
-id: int
-name: string
-username: string
Moderator(string,string):Moderator
1
Vote Class
-voter: EmailAddress
-createdAt: DateTime
-isUpvote: boolean
+DownVotePin(string): int
+FlagAsBad(string,int,enum,string): boolean
+UpVotePin(string): int
Administrator Class
-id: int
-name: string
-username: string
Administrator(string,string):Administrator
BusinessPin Class
-openHours: BusinessHours
-congestion: int
-businessIcon: Icon
+DownVotePin(string): int
+FlagAsBad(string,int,enum,string): boolean
+Pin(string,string,point2D,EmailAddress):Pin
+UpVotePin(string): int
1
EmailAddress Class
-Local-part: string
-domain: string
-createdAt: DateTime
+emailAddress(Local-part,domain): string
1
BusinessHours Class
-openingTimes: array<time>[7]
-closingTimes: array<time>[7]

3.3 Coding Standards & Documentation

- Development tools
    - Virtual Environment
- Front end
    - HTML & CSS
    - Bootstrap
    - JQuery
- Back End
    - Django==2.1.2 (libraries listed)
        - Celery
    - Python 3
- Hosting (TBD, options listed)
    - Google
    - Github pages
    - DigitalOcean
    - Python Anywhere
- Email System (TBD, options listed)
    - Mailchimp
    - Django

4. Versions

Version 1.0 - 2018/11/04

Created the document

- Marek Szymczyk
- Mansi Bhatia
- Max Niebergall

Version 1.1 - 2018/11/20

Editing the document

- Zizheng Huang
- Tayania Siwek-Smith
- Mansi Bhatia
- Sakshi Joshi