Basement Dwellers

KAPP Application
Software Architecture Document

Version <1.2>

Revision History

Date	Version	Description	Author
<10/23/2022>	<1.0>	<First draft> We completely changed our design and the template we used to fit more in-line with other projects.	<Troy, Tanner, Adam, Chris, Thomas>
<10/25/2022>	<1.1>	<Continuing Revisions> Began transferring out project to Github, which we decided would allow us to collaborate easier. We are also continuing to make improvements to our design documents.	<Troy, Tanner, Adam, Chris, Thomas>
<10/30/2022>	<1.2>	<Minor revisions for draft>	<Chris>

Table of Contents

Figures

Software Architecture Document

1. Introduction

1.1 Purpose

This document provides an architectural view of the system. An array of different architectural viewpoints are provided to display all the different aspects of the system. It is primarily used to describe the major architectural decisions on our system.

1.2 Scope

This Software Architecture Document provides an architectural overview of the KAPP Application. The KAPP Application provides information about campus events, classes, and grades to KU students. It will also provide students with an interactive campus map to help students find their way around campus. This product is helpful to allow both incoming and current students to navigate campus and online features the university has to offer.

1.3 Definitions, Acronyms, and Abbreviations

See Glossary, KAPP_glossary_10.30.pdf

1.4 References

1.5 Overview

The rest of this document gives a general overview of KAPP, its software system, and the application's specifications.

2. Architectural Representation

This document presents the architecture as Visual Paradigm models and uses the Unified Modeling Language (UML).

3. Architectural Goals and Constraints

KAPP is developed to work alone and provide students with easy access to maps, notes, assignments, and grades. KAPP is broken into these major components: a app browser (client), application server, and a database server. These components will run collectively to create a seamless user experience. All components will run with iOS and Android.

4. Use-Case View

The Use-Case View is a large overview of the major architectural components of the system. It describes a set of scenarios (use-cases) that interact with eachother within the system. All the use-cases collectively have a centralized function of working uniformly to create a seamless experience for the user.*

*Refer to Use-Case Specifications document for more information → upedu.ucspec.pdf

5. Logical View

5.1 Overview

KAPP is made in two major components: The user interface and the backend application. The user interface is broken down by container. The app container holds the user and user information. It also holds the tab container, which allows the user to choose which tab is selected. The backend stores user information and handles authorization.

5.2 Architecturally Significant Design Packages

5.2.1 Design Model: Packages Diagram

The design module represents the larger package structure and organization of the KAPP system. The package diagram is displayed (Figure 1).

5.2.2 Package Description

Server	
Description:	Main system package. All requests are handled by this package
Corresponding Classes:	Login
Relations:	Main KAPP package dependent of user.
Sub-packages:	Server

Server	
Description:	All information and methods relating to users are handled within this package
Corresponding Classes:	User, Homepage, Header, Tab Container, Tab, Interactive Map, Events, GPA Calculator, Schedule Builder
Relations:	Sub package of server
Sub-packages:	None

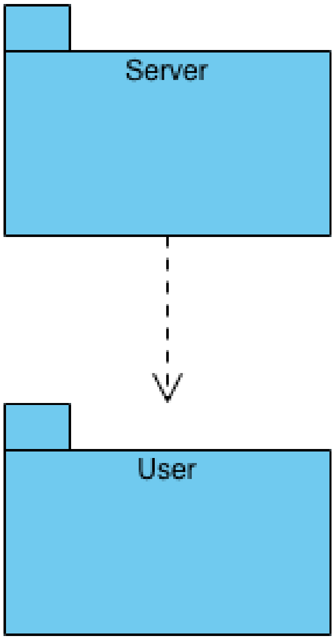

Figure 1: Design Model Package

5.2.3 *Design Model: Class Diagram*

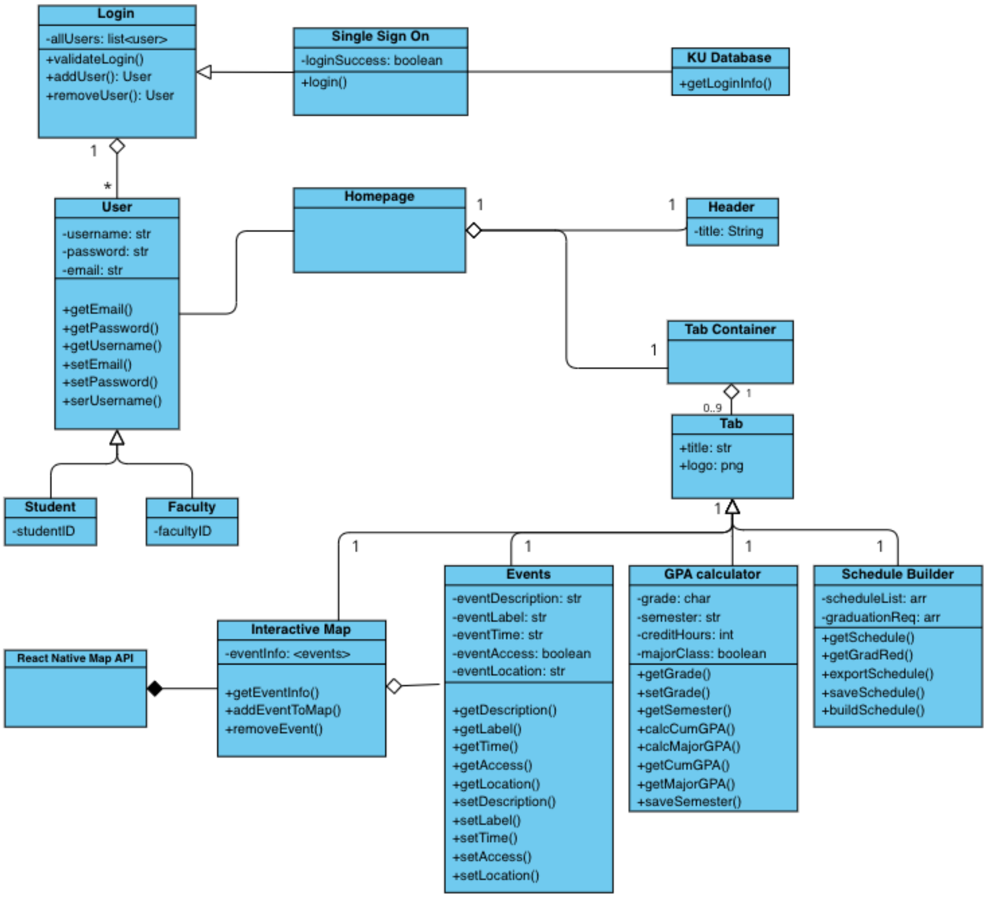

Figure 2: Class Model Diagram

5.3 Design Classes Description

Property	Description
Name	Single Sign On
Description	Handles user sign-in process
Responsibilities	Verifies UserID and Password
Relations	Association with User class
Methods	login(): Begins Log-in Process

Attributes	loginSuccess: Boolean representing successful login
Special Requirements	None

Property	Description
Name	User
Description	User object for handling all app-user interactions
Responsibilities	Contains all user information
Relations	Parent class of Student and Faculty classes. Aggregation to Login. The login can have multiple users. Association to homepage.
Methods	getEmail(): Retrieves email from server getPassword(): Retrieves password from server getUsername(): Retrieves username from server setEmail(): Sets new email setPassword(): Sets new password setUsername(): Sets new username
Attributes	username: private variable for username password: private variable for password email: private variable for email
Special Requirements	None

Property	Description
Name	Student
Description	A type of user that is verified as a student of the University
Responsibilities	Contains information specific to a student
Relations	Generalization > User
Methods	none
Attributes	studentID: Unique KU Student ID
Special Requirements	none

Property	Description
Name	Faculty
Description	A type of user that is verified as faculty of the University
Responsibilities	Contains information specific to a faculty member
Relations	Generalization > User
Methods	none
Attributes	facultyID: Unique KU Faculty ID
Special Requirements	none

Property	Description
Name	Login
Description	Handles Log-in Process into the app
Responsibilities	Collects necessary data from the Student
Relations	Agregation with the User class. There is only 1 login for each user. Parent for KU Single Sign On.
Methods	validateLogin(): gets the user's data addUser(): adds new user to system removeUser(): removes user from system
Attributes	allUsers: list of all users
Special Requirements	none

Property	Description
Name	KU Database
Description	Database containing student account information
Responsibilities	Provide student account information to application
Relations	Association with Single Sign-On
Methods	getLoginInfo(): returns student's login info

Attributes	none
Special Requirements	none

Property	Description
Name	Header
Description	Header for the application
Responsibilities	Display a header on the top of the Application
Relations	Generalization> Homepage
Methods	none
Attributes	title: String containing title for the page
Special Requirements	none

Property	Description
Name	Tab Container
Description	Container for the tabs on the app
Responsibilities	Contain the different tabs/sections of the app
Relations	Parent class of tab. Generalization > Homepage
Methods	none
Attributes	none
Special Requirements	none

Property	Description
Name	Tab

Description	Abstract class representing each tab/each section of the app
Responsibilities	Contains structural information for each tab
Relations	Parent class of Interactive Map, Events, GPA calculator, and Schedule builder
Methods	none
Attributes	Title: name for the tab Logo: logo for the tab
Special Requirements	None

Property	Description
Name	Homepage
Description	Homepage for the app
Responsibilities	Displays Homepage for the app
Relations	Parent class for header and tab container
Methods	None
Attributes	None
Special Requirements	None

Property	Description
Name	Interactive Map
Description	Class representing the Interactive Map functionality of the app
Responsibilities	Displays interactive map for students to view events across campus. Students will be able to have access to a map constrained to campus. An event will pop up around the map, so students can click on it to view information about the event. The map will be made using React Native.
Relations	Generalization > Tab. Aggregation with Events. Composition with React Native Map API
Methods	getEventInfo(): gets event info from event class addEventToMap(): add event with info to map removeEvent(): removes event after it is completed
Attributes	eventInfo: list of events with attributes

Special Requirements	None

Property	Description
Name	Events
Description	Class representing the Events functionality of the app
Responsibilities	Displays events on campus
Relations	Generalization> Tab. Also has Agregation with Interactive Map
Methods	getDescription(): gets description of event getLabel(): gets event name getTime(): gets time of event getAccess(): gets whether event is private or public getLocation(): gets location of event setDescription(): sets description of event setLabel(): sets event name setTime(): sets time of event setAccess(): sets access to event setLocation(): sets location of event
Attributes	eventDescription: Describes what the club is and a general description about the meeting. eventLabel: Title of club/event going on eventTime: The duration of the event from start to stop eventAccess: Whether you must be invited or if it is open to public eventLocation: What building and room the event is taking place in.
Special Requirements	None

Property	Description
Name	GPA Calculator
Description	Class representing the GPA Calculator functionality of the app
Responsibilities	Displays the GPA Calculator
Relations	Generalization> Tab.
Methods	getGrade(): gets grade for given class or semester setGrade(): sets grade for class getSemester(): gets semester for student to edit or view calcCumGPA(): calculates students cumulative GPA calcMajorGPA(): calculates students major GPA getCumGPA(): returns cumulative GPA getMajorGPA(): returns major GPA

	saveSemester(): saves class information for a given semester
Attributes	grade: grade received in the class semester: string for which semester to add a class too creditHours: int for number of credit hours majorClass: boolean whether class is required for major
Special Requirements	None

Property	Description
Name	Schedule Builder
Description	Class representing the schedule builder functionality of the app
Responsibilities	Displays schedule builder
Relations	Generalization> Tab.
Methods	getSchedule(): gets schedule for upcoming semester getGradReq(): gets graduation requirements for user exportSchedule(): exports schedule to user personal calendar saveSchedule(): saves schedule to KAPP buildSchedule(): build new schedule based on graduation requirements
Attributes	scheduleList: array of classes with information gradationReq: array of completed and uncompleted graduation requirements
Special Requirements	None

6. Interface Description

The User-Interface is a tabular model consisting of individual tabs for events, grades, schedule builder, and interactive map. The functions of the interface follow the specifications already stated above. The interface will consist of a login screen that will be then taken to the main homepage. There will be buttons for on the expected functionalities of KAPP, that when pressed will function as expected.

7. Size and Performance

The selected software architecture of KAPP supports use of the application through any typical OS or iOS device. The performance of KAPP is designed to use minimal computing power and memory, since it will be ran primarily on mobile devices.

8. Quality

KAPP follows all software architecture requirements as well as the functional and non-functional specifications as stated in this document.