KFC Single Sign-On Portal

Login

Design Document

CECS 491B, Sec 11

April 17, 2019

Christian Flore-Rogel
013924454

Revision History

Date	**Version**	**Description**
3/18/2019	1.0	First Draft

Table of Contents

1. Introduction 2

2. Login Functionality 3

    2.a. Login Workflow Diagram 3

    2.b. Login Sequence Diagram 5

3. API 6

4. Additional Errors 6

1. Introduction

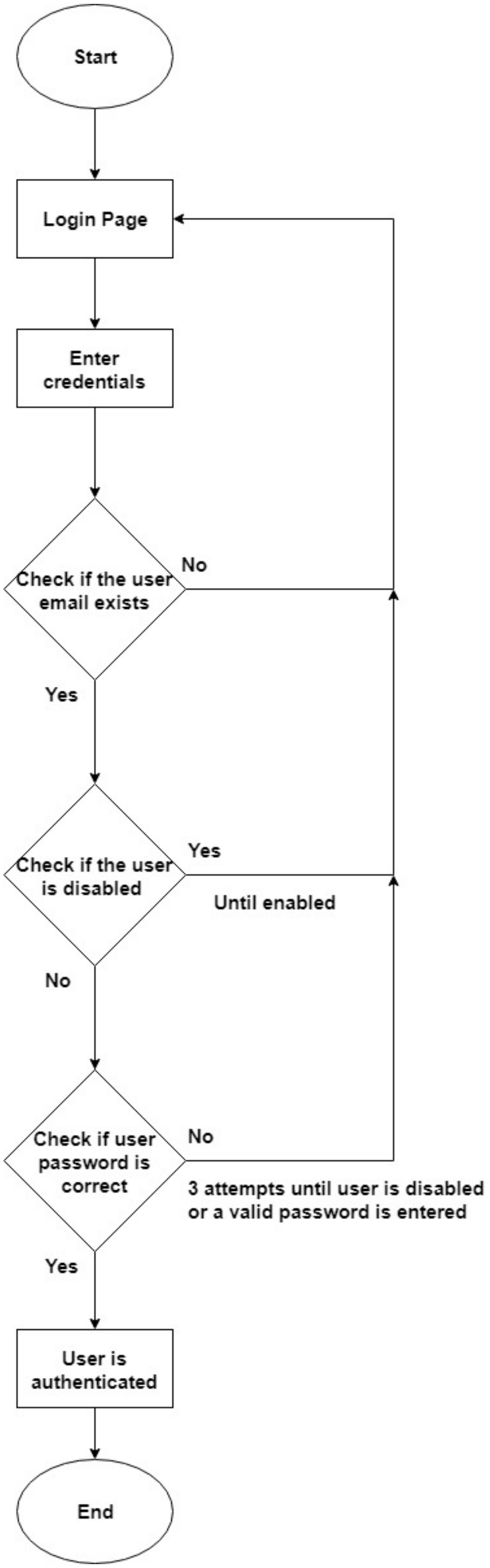
This document purpose is to understand the design for the login feature of the KFC Single-Sign-On application portal. This main point of the login feature is to create a session for the user so he can access to the apps that are published on the dashboard.

In this document a workflow and sequence diagram is shown in order to aide in the understanding of the feature.

2. Login Functionality

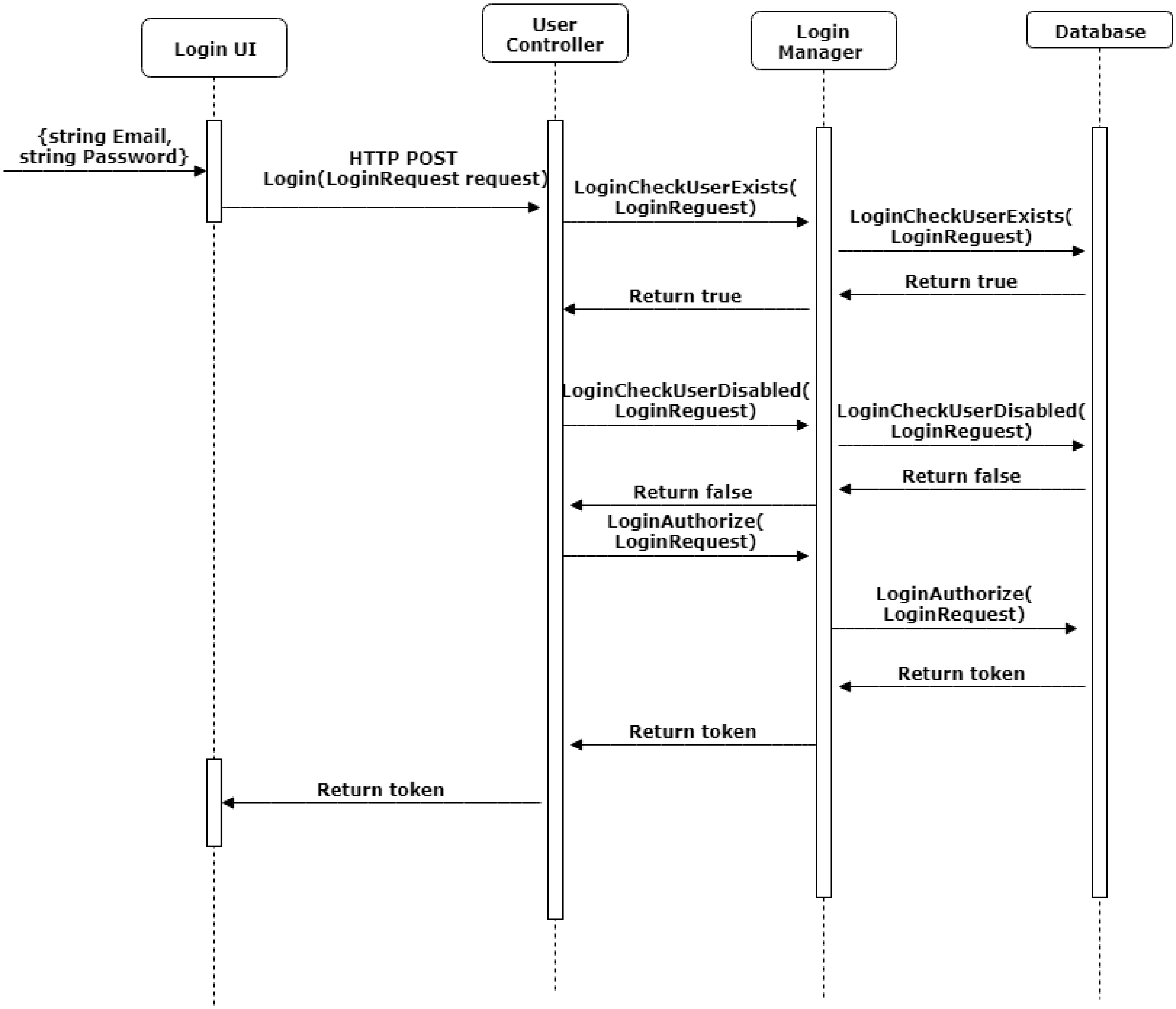
a. Login Workflow Diagram

When a user gets to the home page, the user can click the login page where they can enter the necessary credentials(Email and password) in order to login. If the email doesn't exist, and alert notification is appears that says that the username is invalid. If it does exist then it sequentially checks whether the user is disabled and whether the user's password is correct. If the user is disabled then the user will not be able to login until an administrator enables the user again. If the password is incorrect than, the user will a total of three attempts to login. If the user has 3 incorrect attempts, then the user is disabled and they won't be able to login. If the password is correct, the user officially is logged in and is taken to the dashboard page.


Start
Login Page
Enter credentials
Check if the user email exists
No
Yes
Check if the user is disabled
Yes
Until enabled
No
Check if user password is correct
No
3 attempts until user is disabled or a valid password is entered
Yes
User is authenticated
End

b. Login Sequence Diagram

The sequence diagram show how the UI interacts with the user. In the UI phase, the user inputs their email and password. After the user inputs his information, an axios post call is used in order to connect to a Web API controller so the controller can determine whether the user can login. The post login method in the controller then uses a variety of methods from the manager layer in order to determine whether the use can login. Inside the manager layer, some of the methods connect to the database in order to check whether the information is stored or to read and verify the data. If the information provided is correct, is session token is created and is sent back to the user interface so it can stored in the browser which later enables the user to login.


Login UI
User Controller
Login Manager
Database
{string Email, string Password}
HTTP POST Login(LoginRequest request)
LoginCheckUserExists(LoginReguest)
LoginCheckUserExists(LoginReguest)
Return true
Return true
LoginCheckUserDisabled(LoginReguest)
LoginCheckUserDisabled(LoginReguest)
Return false
Return false
LoginAuthorize(LoginRequest)
LoginAuthorize(LoginRequest)
Return token
Return token
Return token

3. API Documentation

a.

i. Login Request Model

Class: Login Request		
Email	string	Contains inputted email of user
Password	string	Contains inputted password of user

ii. Controller Calls

Title	**Login**	**GetEmail**
Url: api/user	**/login**	**/getemail/{token}**
Method	**POST**	**GET**

Service Response	**Code: 200**
	Content{
	String: Token
	}
		Content{
		String: Email
		}

Error Response	**Code: 400** **Code: 401** **Content:** **{** **Message** **}**	**None**

4. Additional Errors

- If the request entry has an email/password that is not stored in the database, then an HTTP response with a status code of 400 and a message("Invalid Username/Password") will be returned.
- If the request entry has an email that is disabled in the database, the user will not be able to log in and a HTTP response with a status code 401 and a message("User is Disabled") will be returned.