1. Client Requirements:

What our client wants us to make is a journal wherein she could address her personal experiences through writing and pictures chronologically arranged by date. The background to this proposal is that, since she just graduated, she wants to keep all the memories that she kept in her mobile device into her laptop since it is eating memory. She asked us to make it only available only to her laptop since she don’t intend to share it to anyone else but with herself and her loved ones. She did not specify any security check so on our second interview we opened it up to her. These logs should be accessible by panning through by days, months, and years. The client also requests that the overall aesthetic should be as minimalistic as possible with a color palette revolving around black, white, and gray compared to the likes available at the market. In terms of functionality, she mentioned to include the ability to add photos for each entry she would create, she also mentioned adding hashtags to make it more personal. Since a journal should always be kept private, the group decided to add a PIN to prevent other people from prying into the client’s personal entries.

2. Use Case Diagrams:



3. Activity Diagrams:

a.) Create New Entry -



b.) Edit Existing Entry - 

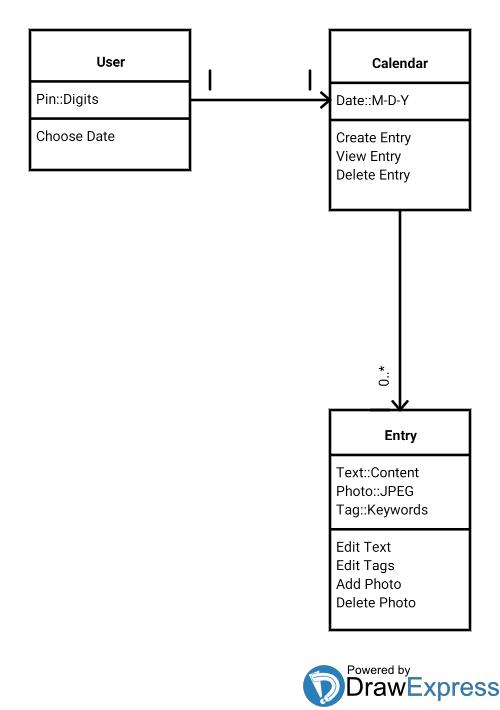
c.) Delete Entry –



d.) Change PIN



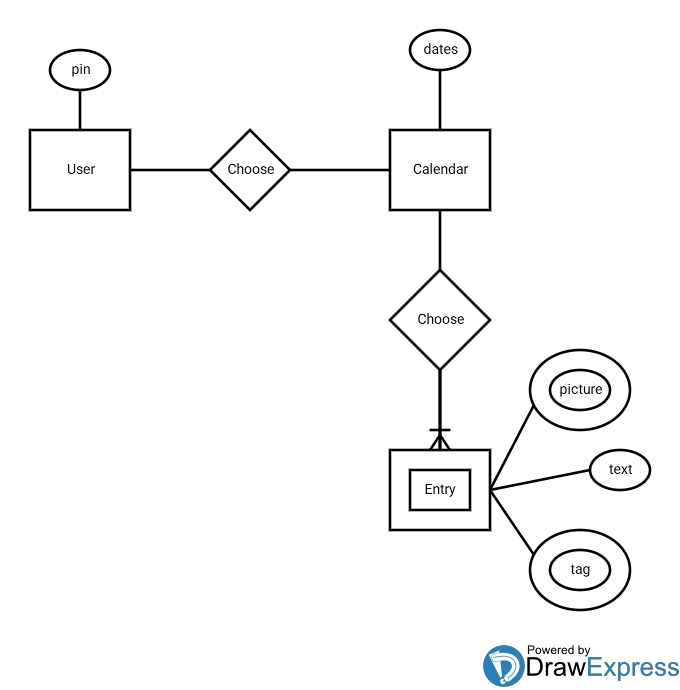
4. Conceptual Class Diagram:



5. Class Diagram:



6. ERD:



7. Unit Tests:

|  |  |  |  |
| --- | --- | --- | --- |
| Class User : verifyLogin(inputPin) | | | |
| Condition | Input | Result | Message |
| inputPin incorrect | …(‘4567’) | return False | Incorrect PIN |
| inputPin are letters | …(‘abcd’) | return False | PIN should be digits |
| inputPin are characters | …(‘!@#$’) | return False | PIN should be digits |
| inputPin is empty | …(None) | None | None |
| inputPin is correct | …(‘1234’) | return True | None |

|  |  |  |  |
| --- | --- | --- | --- |
| Class User : changePin(oldPin, newPin, confirmPin) | | | |
| Condition | Input | Result | Error Message |
| Inputs are letters | …(‘abcd’,’abcd’,’abcd’) | return False | PIN should be digits |
| Inputs are characters | …(‘!@#$’, ‘!@#$’, ‘!@#$’) | return False | PIN should be digits |
| oldPin Incorrect | …(‘4321’,’1234’,’1234’) | return False | Old PIN incorrect! |
| newPin and confirmPin mismatch | …(‘1234’,’4321’,’5678’) | return False | PIN did not match! |
| Inputs are empty | …(None,None,None) | None | None |
| Inputs are correct | …(‘1234’,’4321’,’4321’ | return True | PIN Successfully Changed! |

|  |  |  |  |
| --- | --- | --- | --- |
| Class Date : createEntry(txt,tgs,phts) | | | |
| Condition | Input | Result | Error Message |
| txt is empty | (‘ ‘,’#tag1’,’/home/user..’) | None | Entry cannot be empty! |
| tgs is empty | (‘sample text’,’’,’/home..’) | returns True | None |
| phts is empty | (‘sample text’,’#tag1’,’’) | returns True | None |

|  |  |  |  |
| --- | --- | --- | --- |
| Class Date : dateAvailability() | | | |
| Condition | Input | Result | Error Message |
| Selected date is in DB | self.day = selected\_date | return True | None |
| Selected date is not in DB | self.day = selected\_date | return False | None |

|  |  |  |  |
| --- | --- | --- | --- |
| Class Date: SendData(getWhat) | | | |
| Condition | Input | Result | Error Message |
| getWhat = ‘getText’ | (‘getTexts’) | return db.text | None |
| getWhat = ‘getTags’ | (‘getTags’) | return db.tags | None |
| getWhat = ‘getPhotos’ | (‘getPhotos’) | return db.photos | None |
| other input | (‘testText’) | return None | None |

|  |  |  |  |
| --- | --- | --- | --- |
| Class Entry : addTag(tags,isEdit) | | | |
| Condition | Input | Result | Error Message |
| tags is different and isEdit is False | (‘[#test1]’,False) | self.tag.extend(tags) | None |
| tags is different and isEdit is True | (‘[#test1]’,True) | self.tag = tags | None |
| tags is same and isEdit is False | (‘[#yolo,#lolow]’,’False’) | self.tag = self.tag | None |
| tags is same and isEdit is True | (‘[#yolo,#lolow]’,’True’) | self.tag = self.tag | None |