main.py

```
from kivy.app import App
from kivy.lang import Builder
from kivy.uix.screenmanager import ScreenManager, Screen
from kivy.properties import ObjectProperty
from kivy.uix.popup import Popup
from kivy.uix.label import Label
from database import DataBase


class CreateAccountWindow(Screen):
    namee = ObjectProperty(None)
    email = ObjectProperty(None)
    password = ObjectProperty(None)

    def submit(self):
        if self.namee.text != "" and self.email.text != "" and
self.email.text.count("@") == 1 and self.email.text.count(".") > 0:
            if self.password != "":
                db.add_user(self.email.text, self.password.text,
self.namee.text)

                self.reset()

                sm.current = "login"
            else:
                invalidForm()
        else:
            invalidForm()

    def login(self):
        self.reset()
        sm.current = "login"

    def reset(self):
        self.email.text = ""
        self.password.text = ""
        self.namee.text = ""


class LoginWindow(Screen):
    email = ObjectProperty(None)
    password = ObjectProperty(None)

    def loginBtn(self):
        if db.validate(self.email.text, self.password.text):
            MainWindow.current = self.email.text
            self.reset()
            sm.current = "main"
        else:
            invalidLogin()

    def createBtn(self):
        self.reset()
        sm.current = "create"

    def reset(self):
        self.email.text = ""
        self.password.text = ""
```

```
class MainWindow(Screen):
    n = ObjectProperty(None)
    created = ObjectProperty(None)
    email = ObjectProperty(None)
    current = ""

    def logOut(self):
        sm.current = "login"

    def on_enter(self, *args):
        password, name, created = db.get_user(self.current)
        self.n.text = "Account Name: " + name
        self.email.text = "Email: " + self.current
        self.created.text = "Created On: " + created


class WindowManager(ScreenManager):
    pass


def invalidLogin():
    pop = Popup(title='Invalid Login',
                  content=Label(text='Invalid username or password.'),
                  size_hint=(None, None), size=(400, 400))
    pop.open()


def invalidForm():
    pop = Popup(title='Invalid Form',
                  content=Label(text='Please fill in all inputs with valid
information.'),
                  size_hint=(None, None), size=(400, 400))
    pop.open()


kv = Builder.load_file("my.kv")

sm = WindowManager()
db = DataBase("users.txt")

screens = [LoginWindow(name="login"),
CreateAccountWindow(name="create"),MainWindow(name="main")]
for screen in screens:
    sm.add_widget(screen)

sm.current = "login"


class MyMainApp(App):
    def build(self):
        return sm


if __name__ == "__main__":
    MyMainApp().run()
```

my.kv

```
<CreateAccountWindow>:
    name: "create"

     ame:  ame
    email: email
    password: passw


    FloatLayout:
        cols:1

        FloatLayout:
            size: root.width, root.height/2

            Label:
                text: "Create an Account"
                size_hint: 0.8, 0.2
                pos_hint: {"x":0.1, "top":1}
                font_size: (root.width**2 + root.height**2) / 14**4

            Label:
                size_hint: 0.5,0.12
                pos_hint: {"x":0, "top":0.8}
                text: "Name: "
                font_size: (root.width**2 + root.height**2) / 14**4

            TextInput:
                pos_hint: {"x":0.5, "top":0.8}
                size_hint: 0.4, 0.12
                id:  ame
                multiline: False
                font_size: (root.width**2 + root.height**2) / 14**4

            Label:
                size_hint: 0.5,0.12
                pos_hint: {"x":0, "top":0.8-0.13}
                text: "Email: "
                font_size: (root.width**2 + root.height**2) / 14**4

            TextInput:
                pos_hint: {"x":0.5, "top":0.8-0.13}
                size_hint: 0.4, 0.12
                id: email
                multiline: False
                font_size: (root.width**2 + root.height**2) / 14**4

            Label:
                size_hint: 0.5,0.12
                pos_hint: {"x":0, "top":0.8-0.13*2}
                text: "Password: "
                font_size: (root.width**2 + root.height**2) / 14**4

            TextInput:
                pos_hint: {"x":0.5, "top":0.8-0.13*2}
                size_hint: 0.4, 0.12
                id: passw
                multiline: False
                password: True
                font_size: (root.width**2 + root.height**2) / 14**4
```

```
        Button:
            pos_hint:{"x":0.3,"y":0.25}
            size_hint: 0.4, 0.1
            font_size: (root.width**2 + root.height**2) / 17**4
            text: "Already have an Account? Log In"
            on_release:
                root.manager.transition.direction = "left"
                root.login()

        Button:
            pos_hint:{"x":0.2,"y":0.05}
            size_hint: 0.6, 0.15
            text: "Submit"
            font_size: (root.width**2 + root.height**2) / 14**4
            on_release:
                root.manager.transition.direction = "left"
                root.submit()


<LoginWindow>:
    name: "login"

    email: email
    password: password

    FloatLayout:

        Label:
            text:"Email: "
            font_size: (root.width**2 + root.height**2) / 13**4
            pos_hint: {"x":0.1, "top":0.9}
            size_hint: 0.35, 0.15

        TextInput:
            id: email
            font_size: (root.width**2 + root.height**2) / 13**4
            multiline: False
            pos_hint: {"x": 0.45 , "top":0.9}
            size_hint: 0.4, 0.15

        Label:
            text:"Password: "
            font_size: (root.width**2 + root.height**2) / 13**4
            pos_hint: {"x":0.1, "top":0.7}
            size_hint: 0.35, 0.15

        TextInput:
            id: password
            font_size: (root.width**2 + root.height**2) / 13**4
            multiline: False
            password: True
            pos_hint: {"x": 0.45, "top":0.7}
            size_hint: 0.4, 0.15

        Button:
            pos_hint:{"x":0.2,"y":0.05}
            size_hint: 0.6, 0.2
            font_size: (root.width**2 + root.height**2) / 13**4
            text: "Login"
            on_release:
```

```
                root.manager.transition.direction = "up"
                root.loginBtn()

        Button:
            pos_hint:{"x":0.3,"y":0.3}
            size_hint: 0.4, 0.1
            font_size: (root.width**2 + root.height**2) / 17**4
            text: "Don't have an Account? Create One"
            on_release:
                root.manager.transition.direction = "right"
                root.createBtn()


<MainWindow>:
    n: n
    email: email
    created:created

    FloatLayout:
        Label:
            id: n
            pos_hint:{"x": 0.1, "top":0.9}
            size_hint:0.8, 0.2
            text: "Account Name: "

        Label:
            id: email
            pos_hint:{"x": 0.1, "top":0.7}
            size_hint:0.8, 0.2
            text: "Email: "

        Label:
            id: created
            pos_hint:{"x": 0.1, "top":0.5}
            size_hint:0.8, 0.2
            text: "Created: "

        Button:
            pos_hint:{"x":0.2, "y": 0.1}
            size_hint:0.6,0.2
            text: "Log Out"
            on_release:
                app.root.current = "login"
                root.manager.transition.direction = "down"
```

database.py

```
import datetime


class DataBase:
    def __init__(self, filename):
        self.filename = filename
        self.users = None
        self.file = None
        self.load()

    def load(self):
        self.file = open(self.filename, "r")
        self.users = {}

        for line in self.file:
            email, password, name, created = line.strip().split(";")
            self.users[email] = (password, name, created)

        self.file.close()

    def get_user(self, email):
        if email in self.users:
            return self.users[email]
        else:
            return -1

    def add_user(self, email, password, name):
        if email.strip() not in self.users:
            self.users[email.strip()] = (password.strip(), name.strip(),
DataBase.get_date())
            self.save()
            return 1
        else:
            print("Email exists already")
            return -1

    def validate(self, email, password):
        if self.get_user(email) != -1:
            return self.users[email][0] == password
        else:
            return False

    def save(self):
        with open(self.filename, "w") as f:
            for user in self.users:
                f.write(user + ";" + self.users[user][0] + ";" +
self.users[user][1] + ";" + self.users[user][2] + "\n")

    @staticmethod
    def get_date():
        return str(datetime.datetime.now()).split(" ")[0]
```

save the text file users.txt