INTRO TO Secure & Intelligent Smartphone App

Dr. John Yoon

Mercy College

Situation

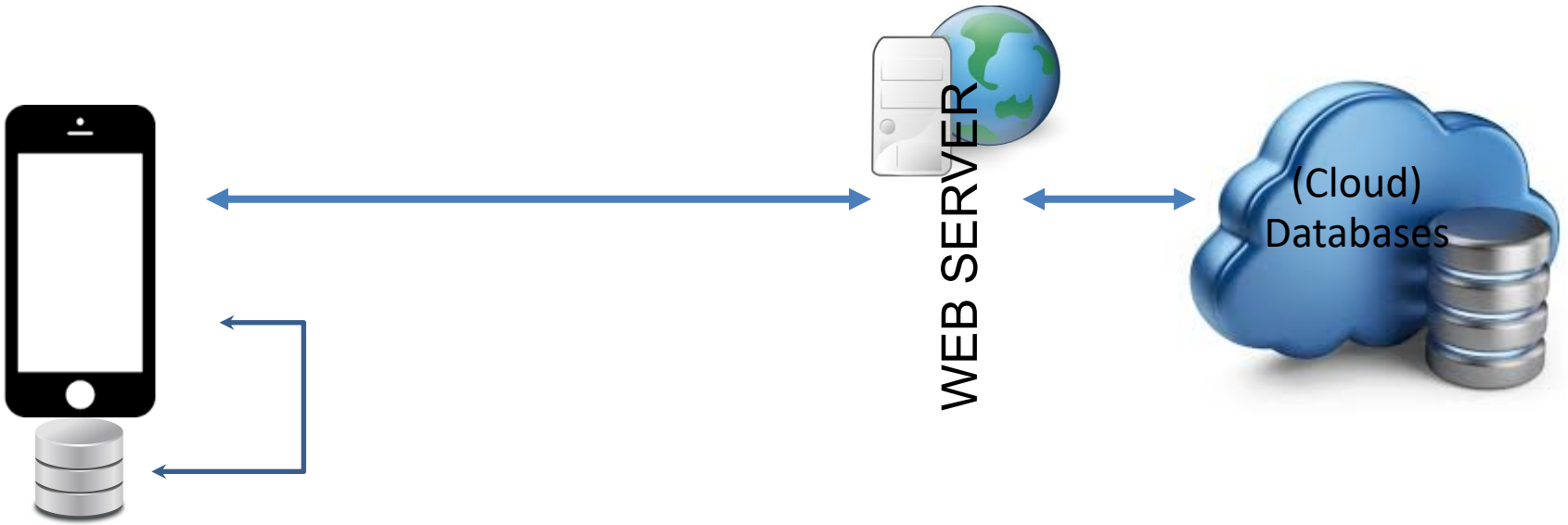
- Mobile devices as
  - Emerging user-interfaces to remote big data servers
  - More users using mobile devices to access servers
  - More attacks on mobile devices
- Smartphone apps are portable
  - Even so, authentication and authorization should not be portable
- Data transmission increases
  - If any deep learning is needed
  - Data analytics may be available on remote servers

Issues

Data transmission between mobile devices and remote servers

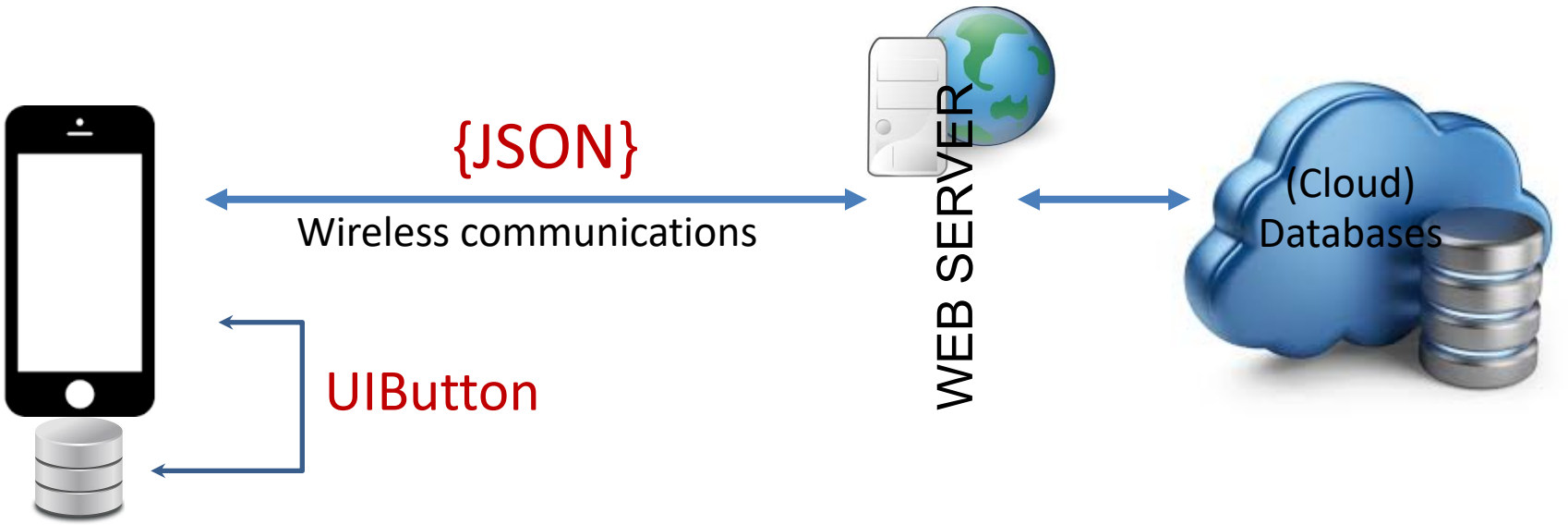
- User authentication
  - Determined on a local device only, or from a remote server?
- Apps authorization
  - Made by a local device decision, by a remote server?
- Outcomes of mobile apps
  - Stored on a local device, or posted to remote servers?
- Logs of app activities
  - Once attacked, one of the important evidence pieces is missing
  - Saved on local device, or posted to remote servers?

Data Used in Mobile Apps



- Data available in the local device, e.g., SQLite3
- Data remotely transmitted from webserver
  - Back-end database, MySQL
  - In the middle, Apache server running PHP

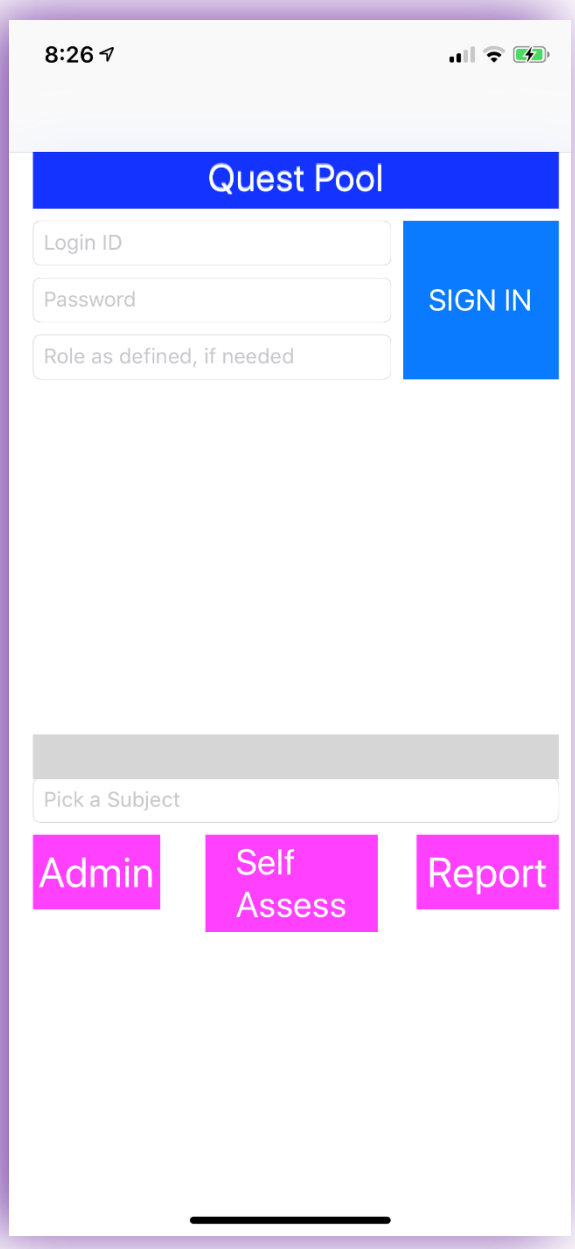
Data Securely Used in Mobile Apps

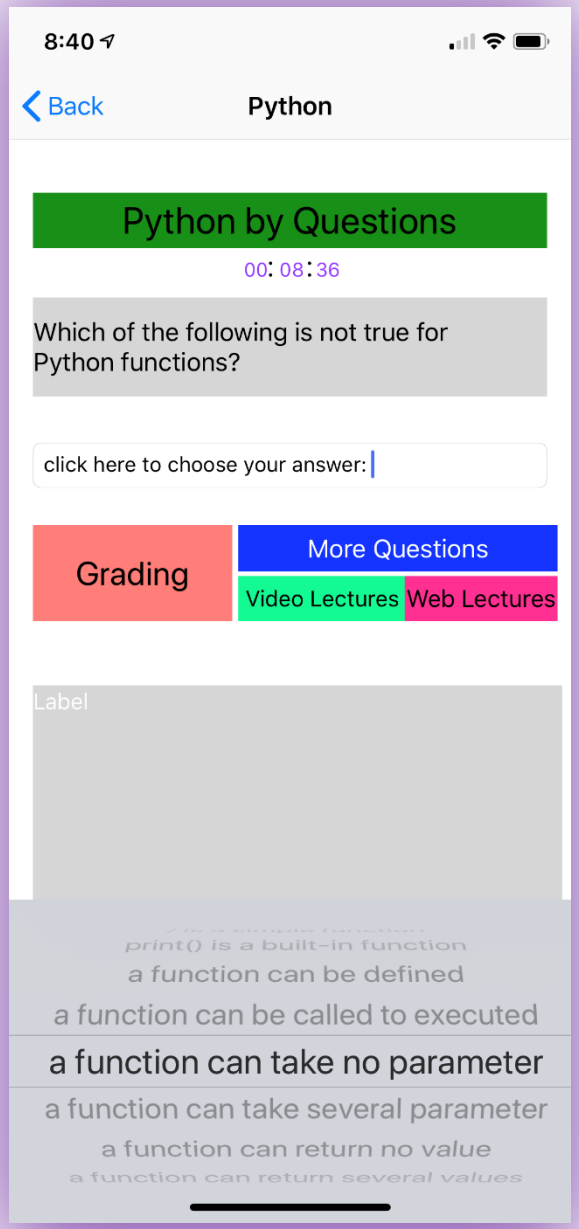


Consider 2 Types

- Data available in the local device, e.g., SQLite3
  - Access Control on UI
- Data remotely transmitted from webserver
  - Control on JSON

Concept Proving





Selecting Questions

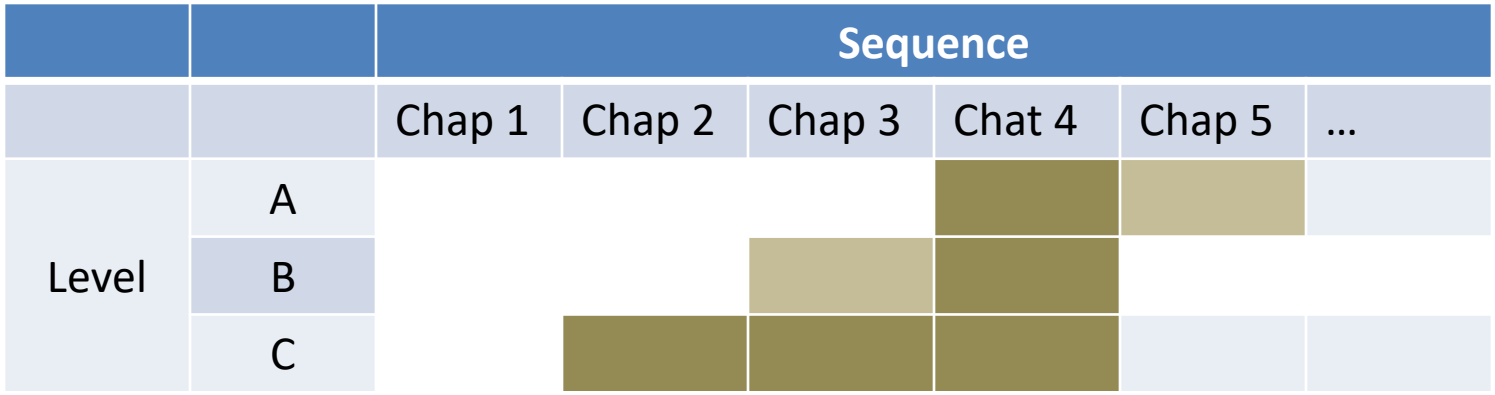
- Questions
  - Selection by combining 1) Randomization, 2) Progressive promotion, and 3) User's level and performance
- User's performance
  - Recorded and stored in another database
- Selections
  - Ex) for three levels of difficulties in chapters, 1,2,3,…, gradual upgrading and progressing lessons.
  - Ex) Consider the below:

		sequence					
		Chap 1	Chap 2	Chap 3	Chat 4	Chap 5	…
Level	A						
	B						
	C						

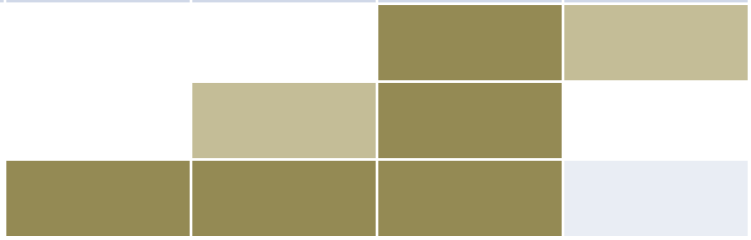
		Chap 1		Chap 3	Chat 4	Chap 5	…
Level	A						
	B						
	C						

		sequence					
		Chap 1	Chap 2	Chap 3	Chat 4	Chap 5	…
Level	A						
	B						
	C						

		Sequence					
		Chap 1	Chap 2	Chap 3	Chat 4	Chap 5	…
Level	A						
	B						
	C						


Sequence
Chap 1
Chap 2
Chap 3
Chat 4
Chap 5
…
Level
A
B
C

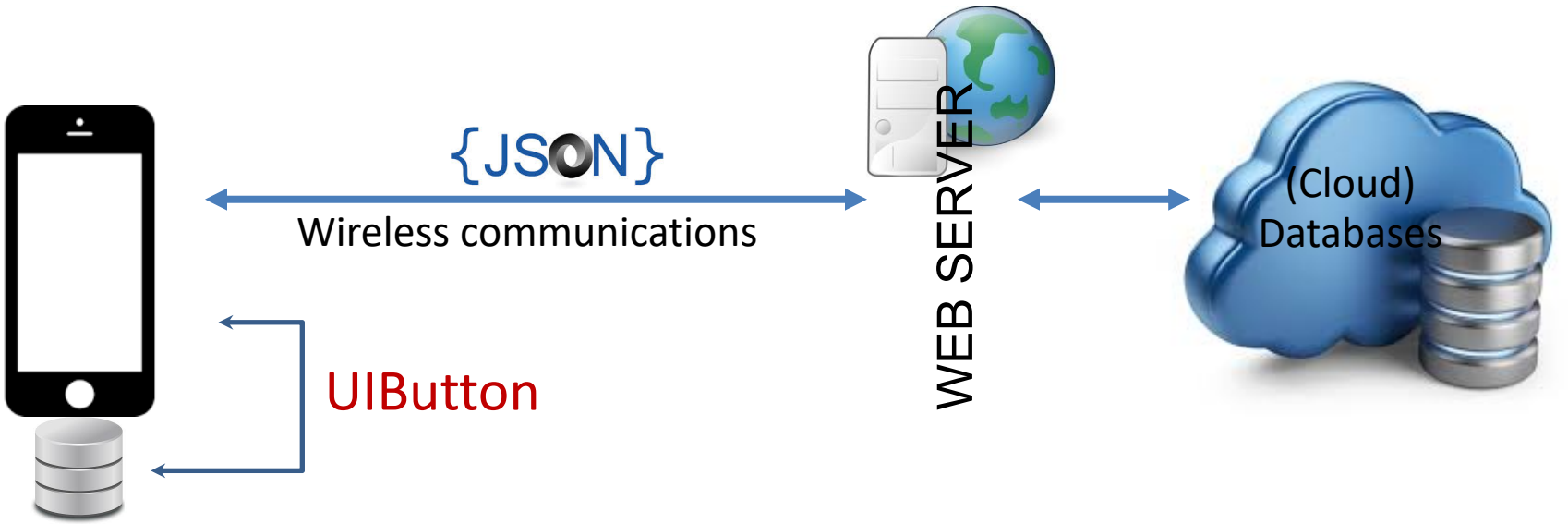
		Sequence					
		Chap 1	Chap 2	Chap 3	Chat 4	Chap 5	…
Level	A						
	B						
	C						



Access Control

- Determined by user's credentials (provided from the remote database)
- Authentication
  - User's login and password
  - UI button to enable/disable
- Authorization
  - Authorize a user to access an App of questions
  - Ex) Authorize John to access the questions on Python
  - UI options to reconstruct for a user to pick

Access Control on UI

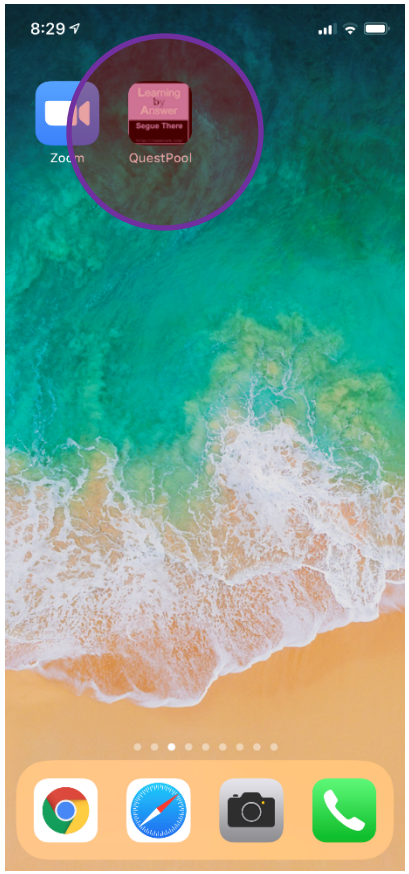


Using **Interface Builder** Outlet/Action in Swift

Simple Implementation

```
@IBOutlet var btnDBAccess: UIButton!
@IBAction func btnDBAccess {_ sender: Any) {
   if idPassVerificier(id, password) {
      btnDBAccess.isEnabled = true
   } else {
      btnDBAccess.isEnabled = false
   }
}
```

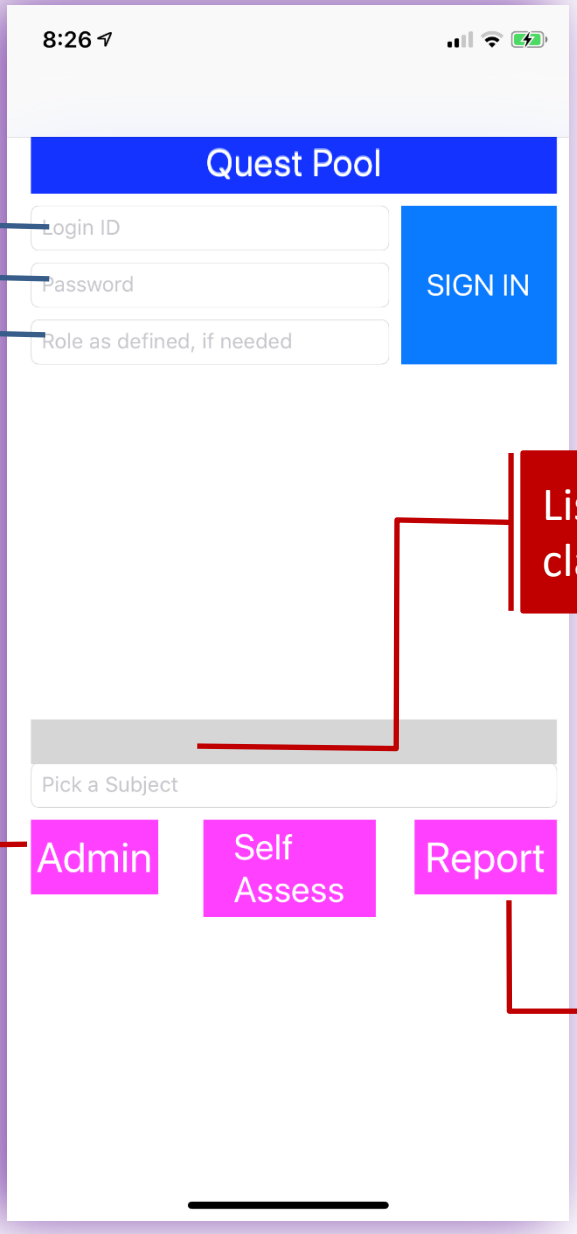
Implementation



Login

Password

Role

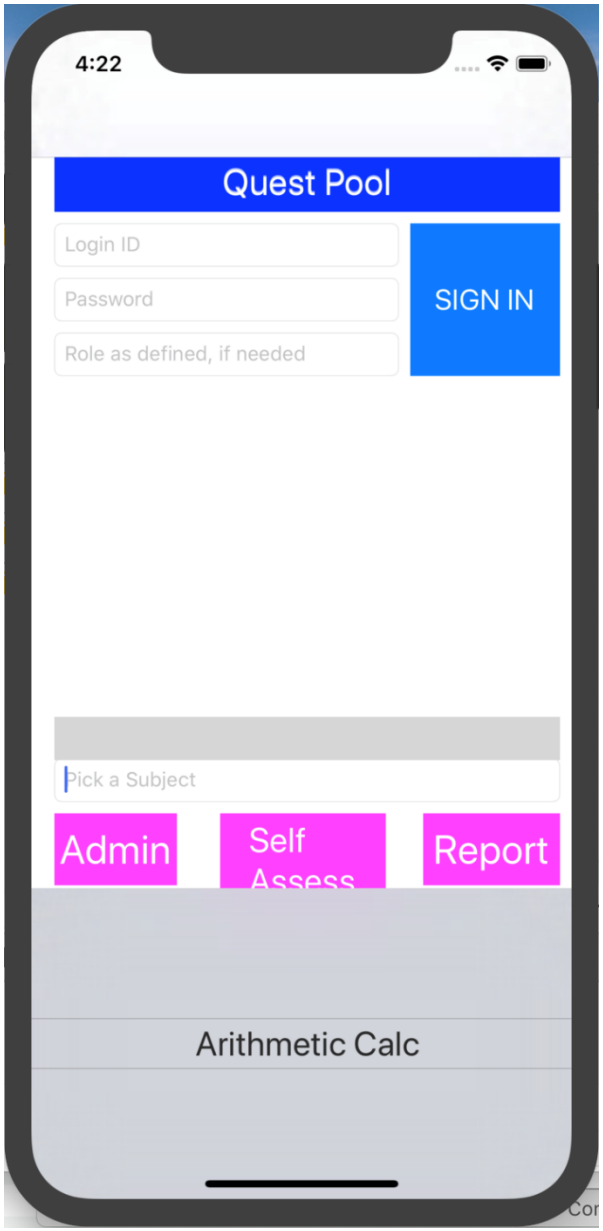


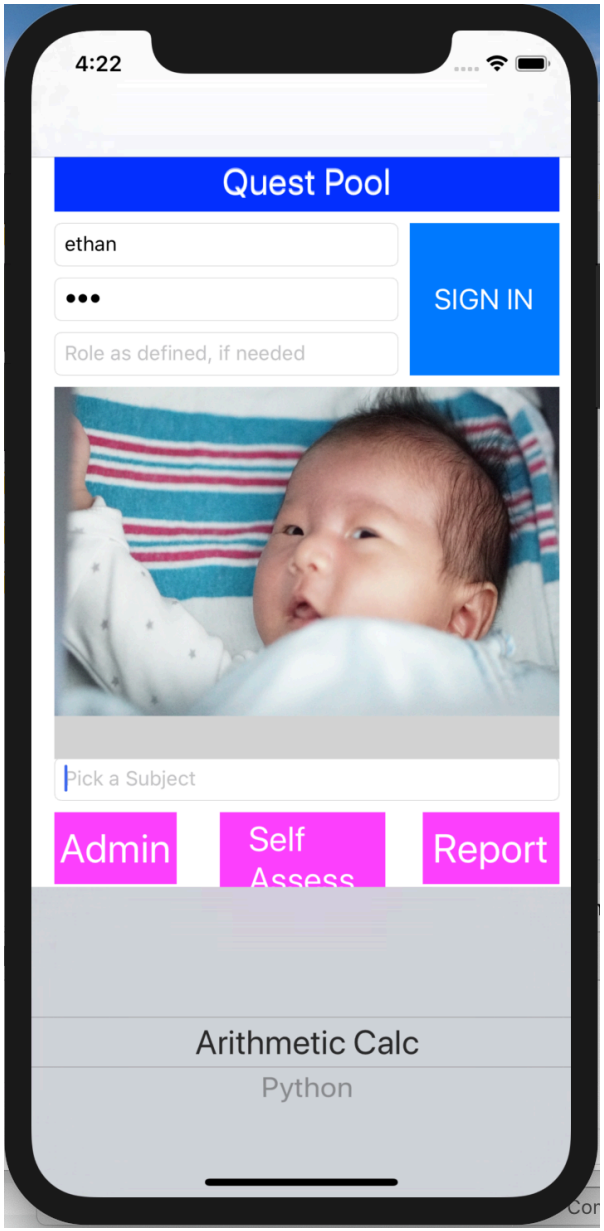
List of authorized classes

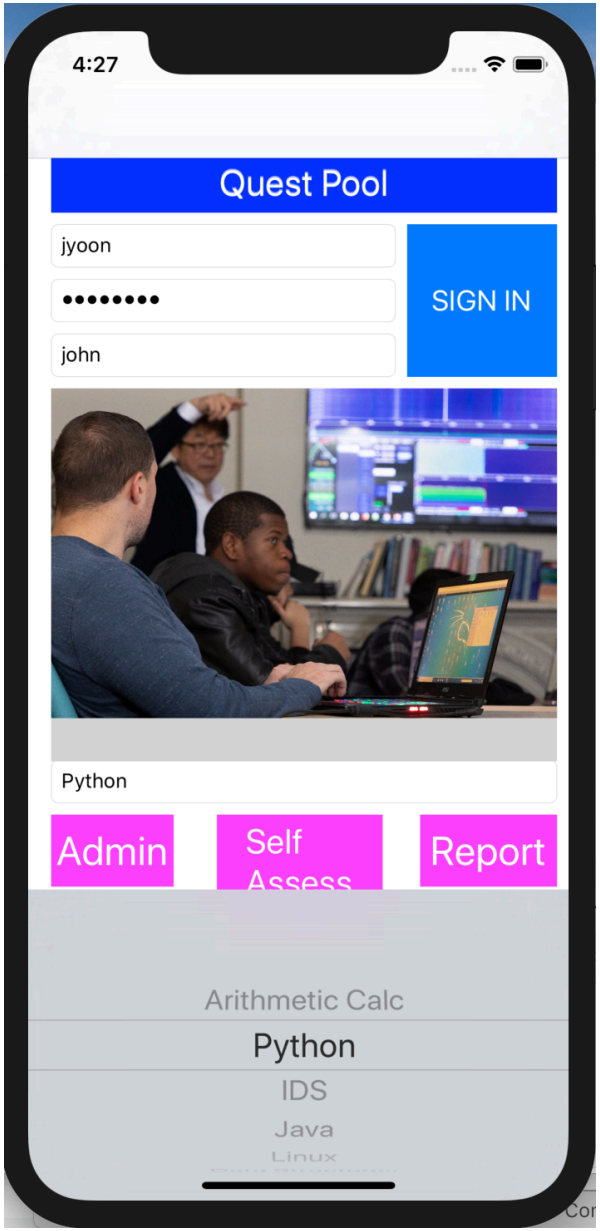
Student management from a class teacher

Class performance analytics

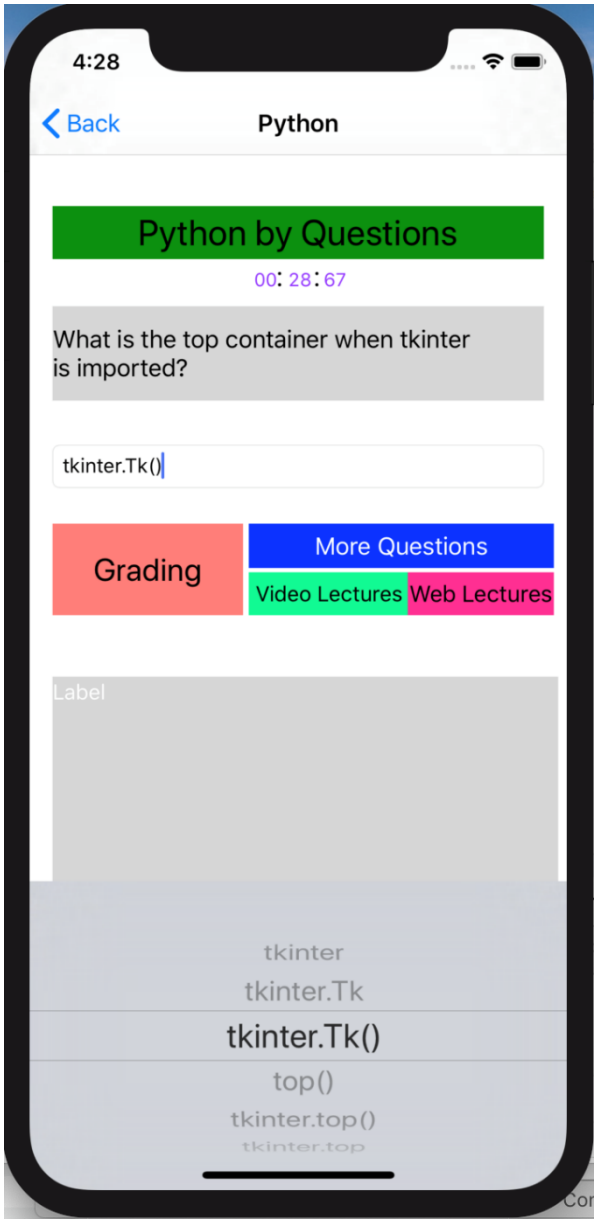
UI Pickerview Created on Users

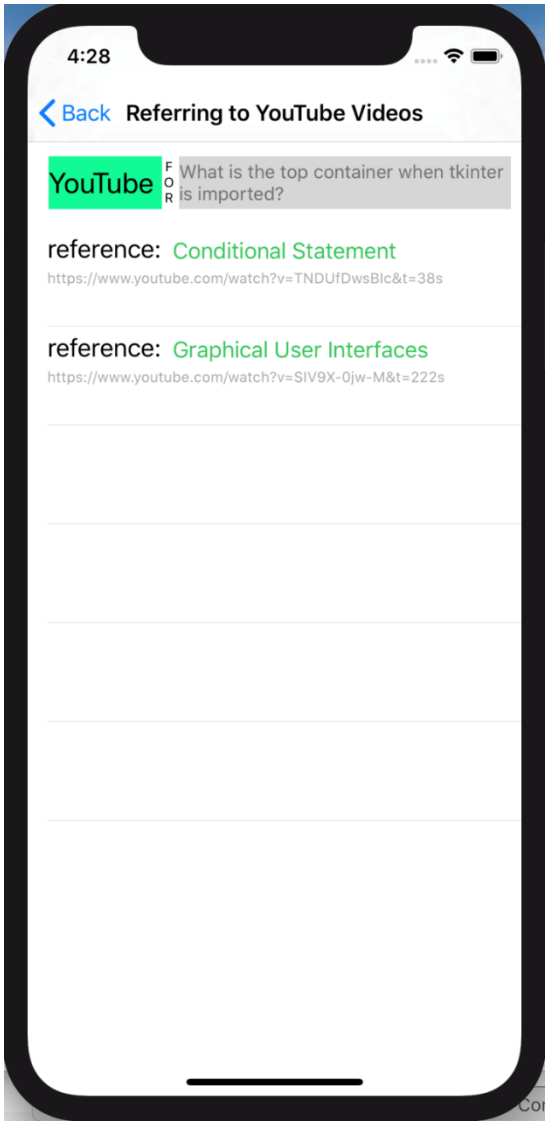


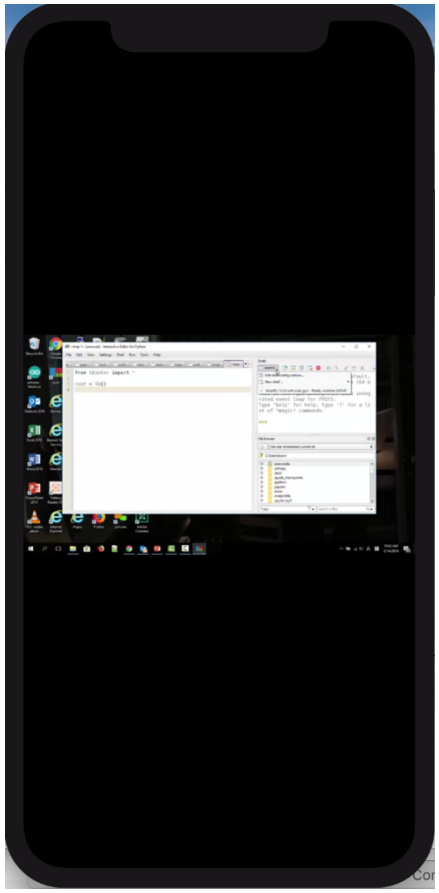




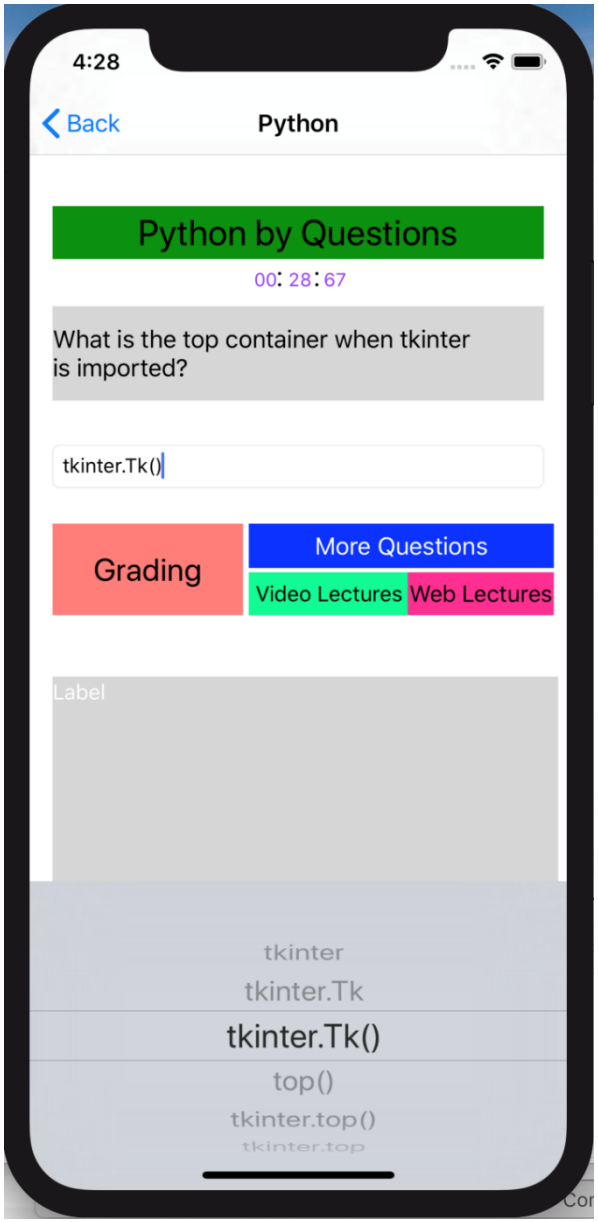
Addition Study from Youtube

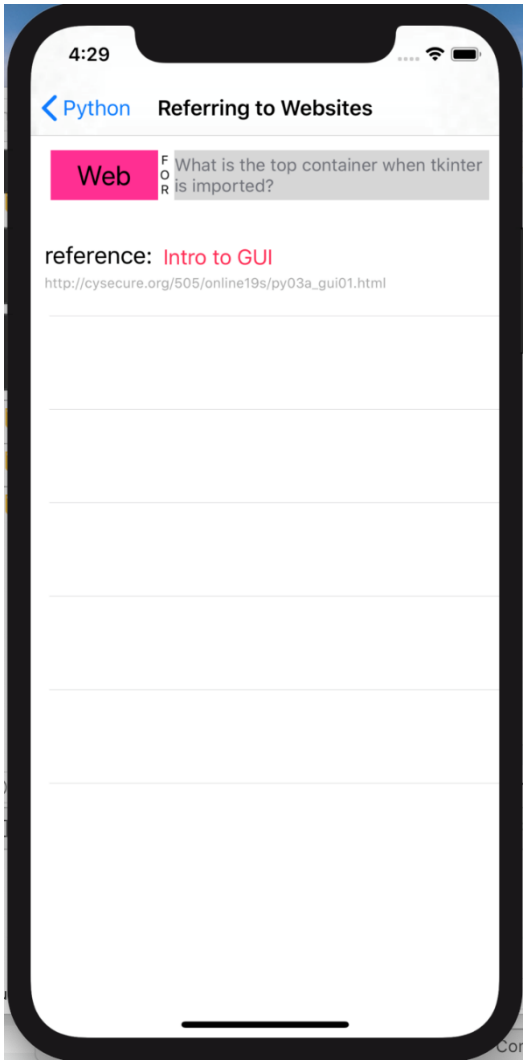


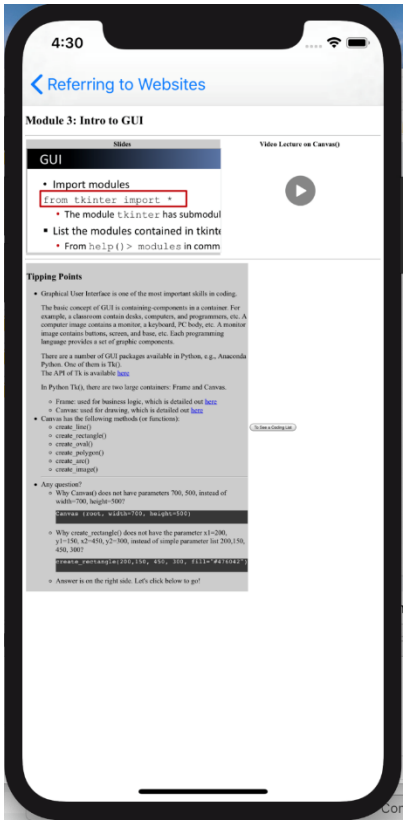




Addition Study from Web Server







Traditional Approach

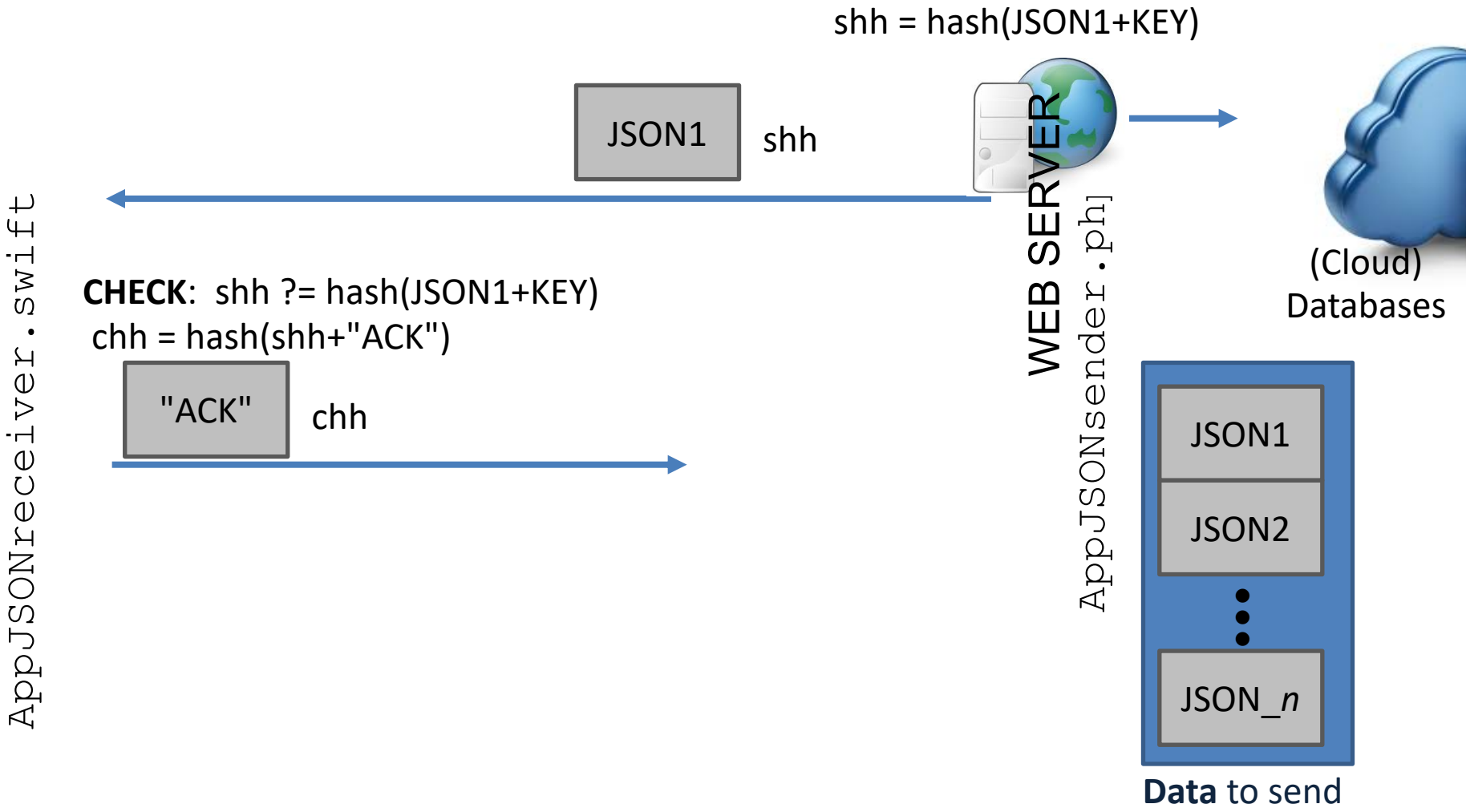
- Cryptographic Approach
  - Encrypt JSON data
  - Works well for each single transaction
  - Once data compromised, data confidentiality is losing and spoofing is still possible

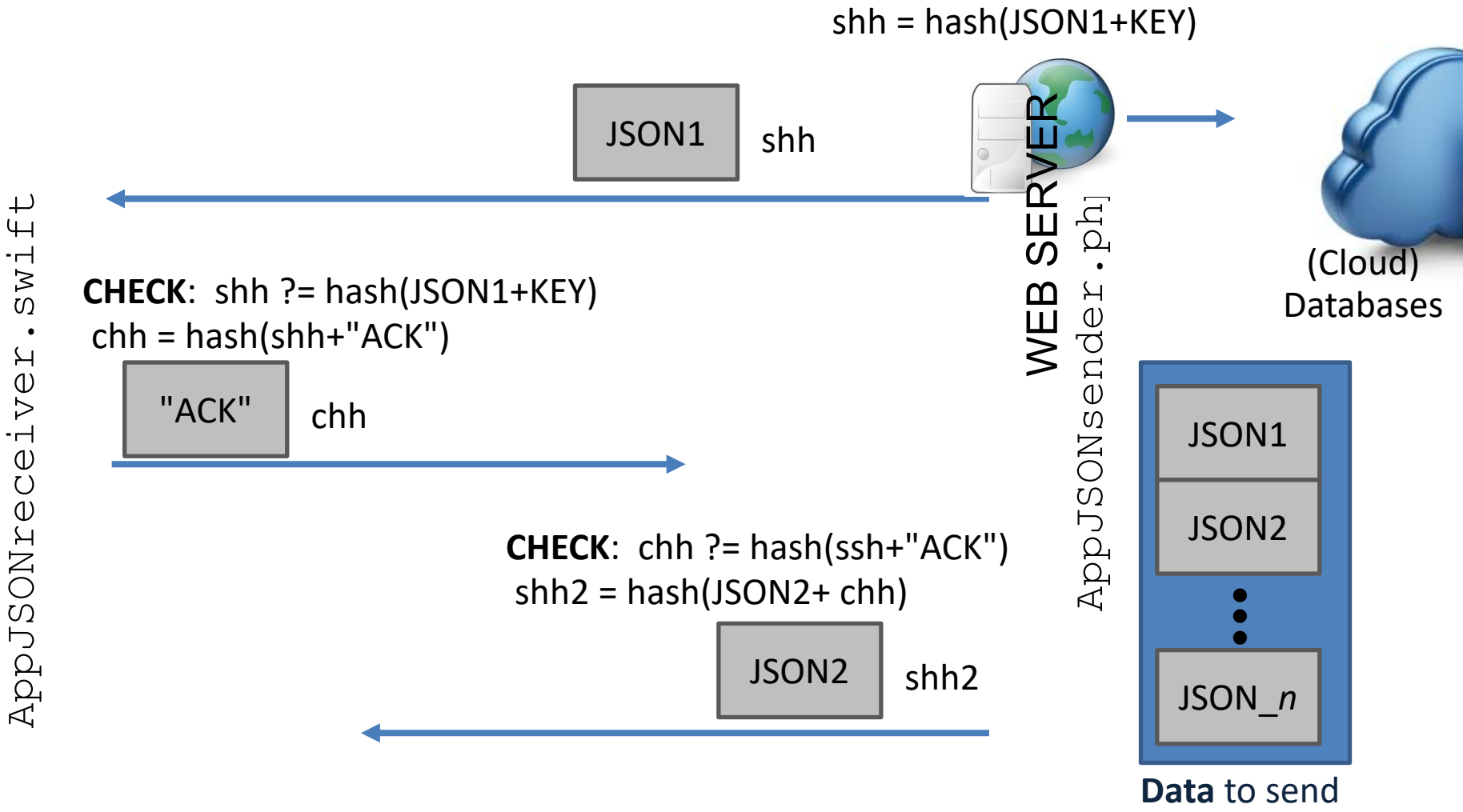
Data Spoofing is Worse in

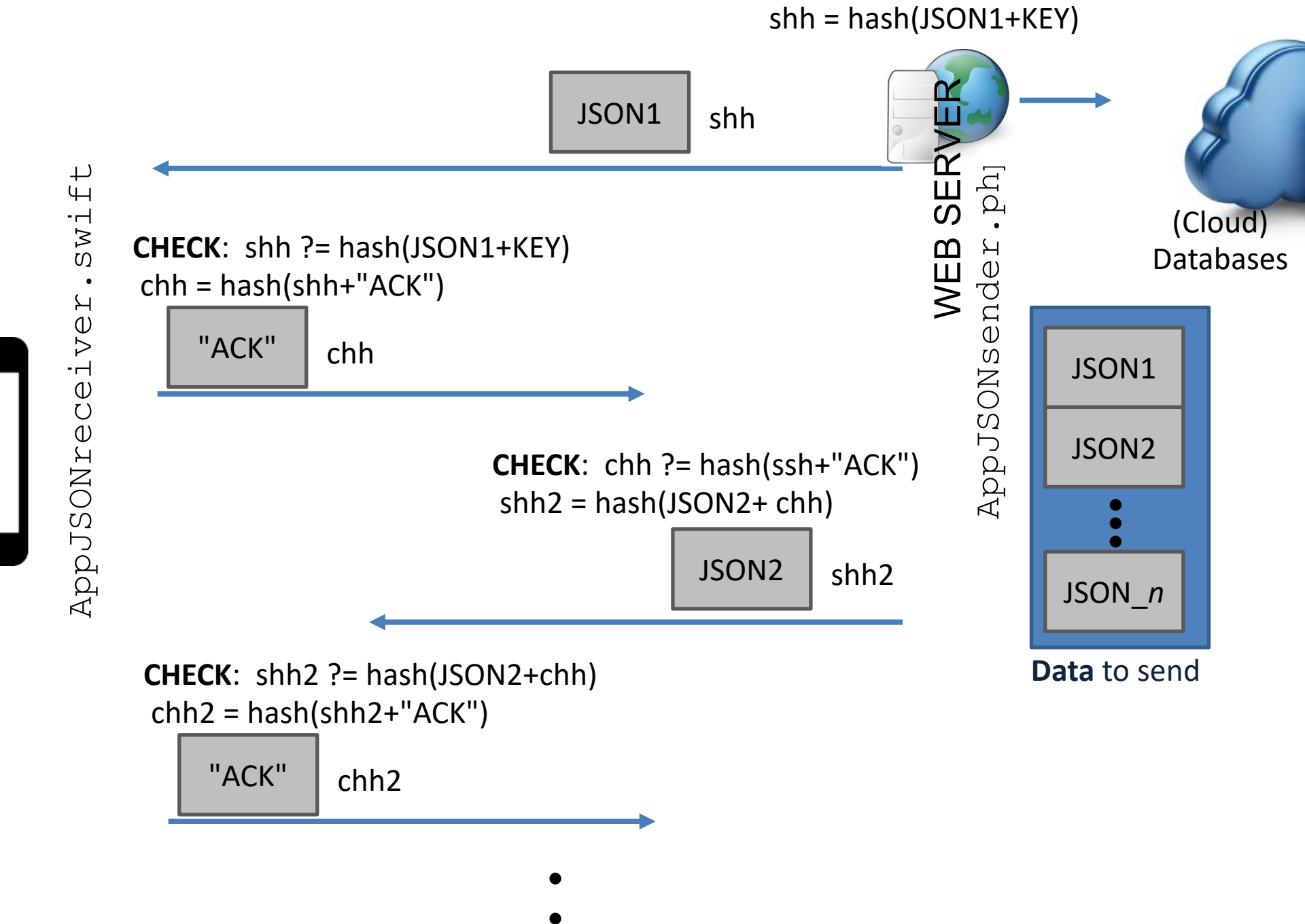
- Data associated with long-duration data transmission
  - Multiple data download and upload
  - Multiple mobile devices and servers
  - Data is linearly incremented

- Proposed Approach
  - Block-chaining of JSON
  - Hashing blocks in chaining

shh = hash(JSON1+KEY)
JSON1
shh
WEB SERVER
AppJSONsender.php
(Cloud)
Databases
JSON1
JSON2
JSON_n
Data to send
AppJSONreceiver.swift


shh = hash(JSON1+KEY)
JSON1
shh
WEB SERVER
AppJSONsender.php
(Cloud)
Databases
CHECK: shh ?= hash(JSON1+KEY)
chh = hash(shh+"ACK")
"ACK"
chh
AppJSONreceiver.swift
JSON1
JSON2
JSON_n
Data to send


shh = hash(JSON1+KEY)
JSON1
shh
WEB SERVER
AppJSONsender.phj
(Cloud)
Databases
AppJSONreceiver.swift
CHECK: shh ?= hash(JSON1+KEY)
chh = hash(shh+"ACK")
"ACK"
chh
JSON1
JSON2
JSON_n
Data to send
CHECK: chh ?= hash(ssh+"ACK")
shh2 = hash(JSON2+ chh)
JSON2
shh2


shh = hash(JSON1+KEY)
JSON1
shh
WEB SERVER
AppJSONsender.php
(Cloud)
Databases
AppJSONreceiver.swift
CHECK: shh ?= hash(JSON1+KEY)
chh = hash(shh+"ACK")
"ACK"
chh
JSON1
JSON2
JSON_n
Data to send
CHECK: chh ?= hash(ssh+"ACK")
shh2 = hash(JSON2+ chh)
JSON2
shh2
CHECK: shh2 ?= hash(JSON2+chh)
chh2 = hash(shh2+"ACK")
"ACK"
chh2

Concluding Remarks

- Secure mobile apps can be implemented
- Authentication and authorization are controlled and managed by a backend server
- Data can be transferred securely between mobile devices and servers

- Cybersecurity classes apps are developed
  - Question-driven student learning
  - Questing-driven student performance analysis