Architecture & Authentication

Purpose fo the mobile app

- To provide a seamless user interface
- Cache (what?) data locally so that the app isn't useless when the user looses internet connection
- To encourage user engagement

Purpose of the server

- To authenticate users
- To provide access to data (via API)
- To provide the search functionality
- To store user state (?)

Server-side language options

Python	**Java**	**Javascript**
+ team familiarity	+ Lots of performance / profiling tools	+ Pretty much the same as python, but without the familiarity
+ speed of development	+ Application might scale better?	+ Lots of learning resources
- Could get harder to maintain as the app grows	- Lack of team expertese	? Faster than Python, but not as fast as java ?
	- Slower to develop (learning curve, boilerplate code)	- Lack of team expertese
	? Better performance	

API Type options

- `Restful` ... what we all know and love
- `GraphQL`
  - The "new kid", lots of hype
  - Could be interesting.
  - Allows you to specify in your request exactly which fields get returned, so you can

send / recieve less data
    - ...however we probably won't have THAT much data, so this won't affect us so much
    - https://www.youtube.com/watch?v=PEcJxkylcRM

Data Storage Options

SQL (MySQL, Amazon RDS)	**No-SQL (Mongo, Redis, Dynamo)**	**Text-files**
+ Familiar	+ Faster simple queries	+ Can be up-and-running immediately
+ Faster for complex queries across tables, with joins etc	+ Schema can evolve without having to rebuild the database	+ Simplify back-end architecture
+ More naturally represents different types of relationships across data	- Might be more difficult to query across tables	- Not a realistic, production-read solution
- Have to decide schema in advance and stick to it		
- (Much) slower for simple queries than NoSQL		

Although we've been told we won't get many marks for back-end stuff, I am a little concerned using the text-file approach may still be held against us? Thoughts?

Useful frameworks / libraries per server-side language

Python

API routing:

- `Flask` (flask-restful),
- `Falcon` (looks simpler to use than flask, but might be trickier to manage user logins?)

Hashing Passwords

- `Bcrypty` (hashes, salts and validates passwords)

Java

`Jersey` is a framework that seems reccomended for Rest APIs

Maybe also consider `Swing`?

Javascript

`Express` is probably the most common library used for this sort of thing, however `koa` is a more modern alternative made by the people who originally made Express.

Backend PaaS / IaaS Options

AWS	Heroku	Firebase
+ We're all familiar	???	+ Made by google, with mobile dev in mind
+ Provides everything we need	? Could warrent looking into, but don't know if anyone on the team has experience	+ Looks like it makes a lot of things simple (see link in auth section)
		- It's another something new to learn, and we don't have a lot of time

Azure / google cloud not included, figuring that they'd be much the same as AWS, but with less team familiarity

Authentication options

OAuth	Roll-Your-own security	Firebase Authentication
+ Common	+ Fewer layers of abstraction	+ Aims to ease the process, handles a lot of the comlexity
+ Flexible	+ Many libraries to help (py-bcrypt is often reccomended)	- Adds other complexity, as you may need to learn the ecosystem?
+ Secure	- Will only have data users explicitly type in for us	
+ Can grab user data while authenticating	- Less 'welcoming' UX	
- More complecated process than roll-your-own	- Additional security concerns	? Part of a larger ecosystem

Resouces:

Firebase Authentication how-to

Youtube: Oauth overview

Python/Flask specific:

Grinberg: Oauth and Flask

Bcrypt and Flask-Login

Flask-Login Home

Additional potential painpoints

- SSL is required: https://stackoverflow.com/questions/25860693/are-oauth2-client-apps-required-to-have-ssl-connection