

**Snap Challenges**

Photography Gamified App

**JACOB ALLEN**

**19003931**

**UXCFXK-30-3**

**Digital Systems Project**

Text

Description automatically generated

# **Table of Contents**

# **Abstract**

# **Acknowledgements**

# **Table of Figures**

# **Table of Tables**

# **1 – Introduction**

**1.1 – Project Timeline (Gantt Chart)**

**2021**

**2022**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | OCT | NOV | DEC | JAN | FEB | MAR | APR |
| Research |  | |  |  |  |  |  |
| Requirements |  |  |  |  |  |  |  |
| Database Design |  | First Draft Completed – 19/11/21 |  |  |  |  |  |
| API Design |  | First Draft Completed – 20/11/21 |  |  |  |  |  |
| UI Design |  |  |  | |  |  |  |
| API Implementation |  |  | |  |  |  |  |
| Database Implementation |  |  | |  |  |  |  |
| Front End Implementation |  |  |  | | |  |  |
| Internal Testing (Own) |  |  |  |  |  |  |  |
| External Testing (User) |  |  |  |  |  |  |  |
| Finalise Report |  |  |  |  |  |  |  |

|  |  |
| --- | --- |
| **Legend** | |
| Report Tasks |  |
| Design Tasks |  |
| Development Tasks |  |
| Testing Tasks |  |

**1.2 – The problem**

There are many existing popular photography sharing platforms, Instagram, 500px and Flicker to name a few. However, none of these sites offer an engaging way to get more images shared and often do not include detailed information about the photographs on the platform. Snap Challenges will attempt to resolve the gaps left by these sites by offering an engaging and informative photo sharing experience.

Photographers often experience burn out from their art, this can be for a variety of reasons, such as:

* Not being able to reach new locations.
* Not experimenting with styles outside of their comfort zone.
* High cost of new gear.
* A lack of engagement with the content they share.

Whilst Snap Challenges will not claim to be able to entirely eliminate these burn out factors, Snap challenges will attempt to mitigate this via the use of gamification. The app will provide users with challenges created by users and admins. These challenges will be regular and achievable yet challenging. This approach should result in more engagement with the platform than a typical photography sharing platform.

**1.3 – Why it’s important**

One of the main ways to improve at photography is to just get out there in the field, and practice. Snap Challenges gives the user a reason to get out and do this practice. This will hopefully lead to an improvement in their photography which benefits the whole community as we get to see better photos.

The app will show camera settings such as:

* Make
* Model
* Aperture
* Shutter Speed
* ISO
* Focal Length

Providing users with this level of detail about a photograph provides a good insight in to how the photo was taken. This will mean that the community can collaboratively improve their photography simply by observing others work.

**1.4 – Aims and Objectives**

AIMS:

1. To develop a platform independent app to gamify photography via the use of challenges.
2. To minimise burn out amongst photographers using these challenges to engage them with the app and their hobby.
3. To educate the users of the app by providing information about the camera settings used to achieve the resulting photo.

OBJECTIVES:

1. Research gamification benefits and drawbacks, and other times it has been used in the photography field.
2. Research React Native in comparison to Progressive Web Apps (PWAs).
3. Research data storage solutions MYSQL vs NOSQL.
4. Design and develop the database schema.
5. Design and develop the Python Flask API.
6. Design and develop the front-end client as either a React Native app or PWA.
7. Test the system from both a technical and user perspective.
8. Produce a report to summarise findings, research, implementation, and testing stages of development.

**1.5 – Potential ethical and legal issues**

The system will handle user data therefore it will need to conform to the General Data Protection Regulation (**GDPR**). There is no intent to store user passwords in the database as it will be more beneficial for the user to log in via an existing account they have on another platform. This will be achieved via OAuth2.

# **2 – Literature Review**

# **3 – Requirements**

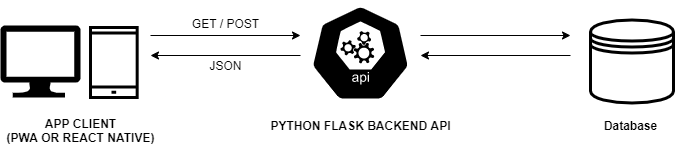
**3.1 Functional Requirements:**

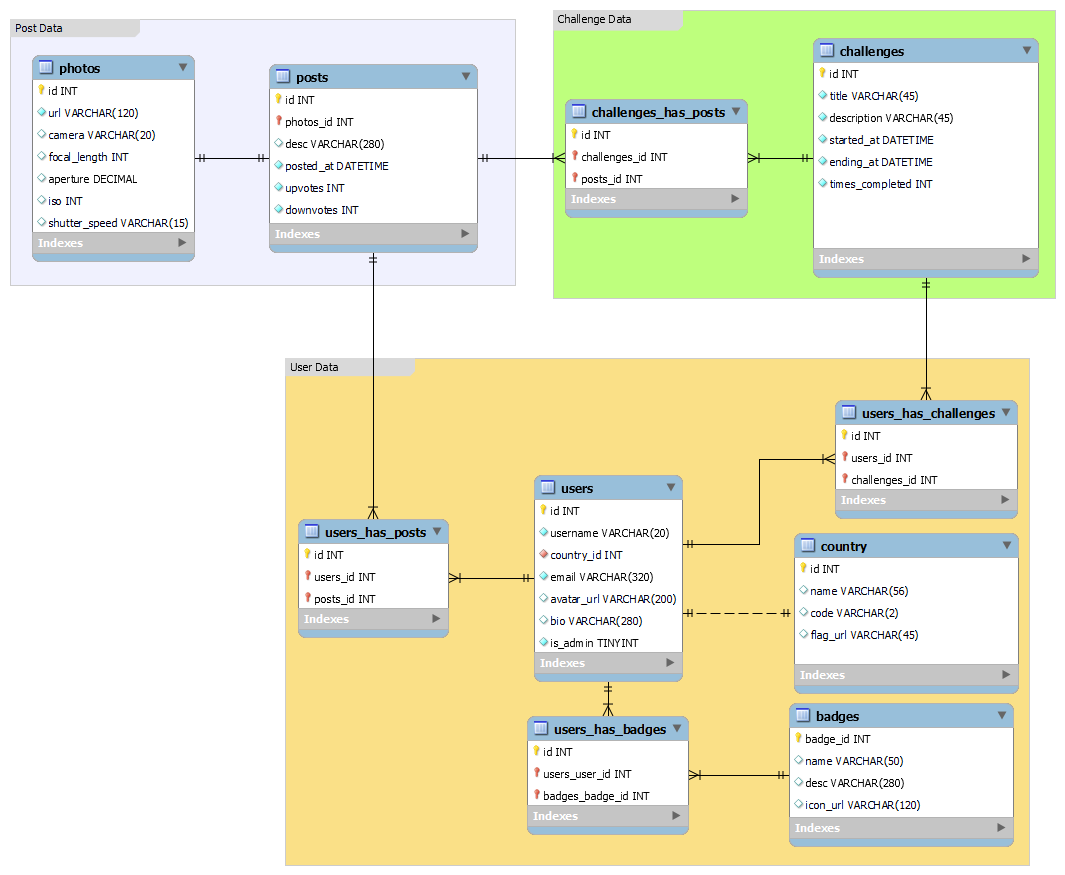
**API:**

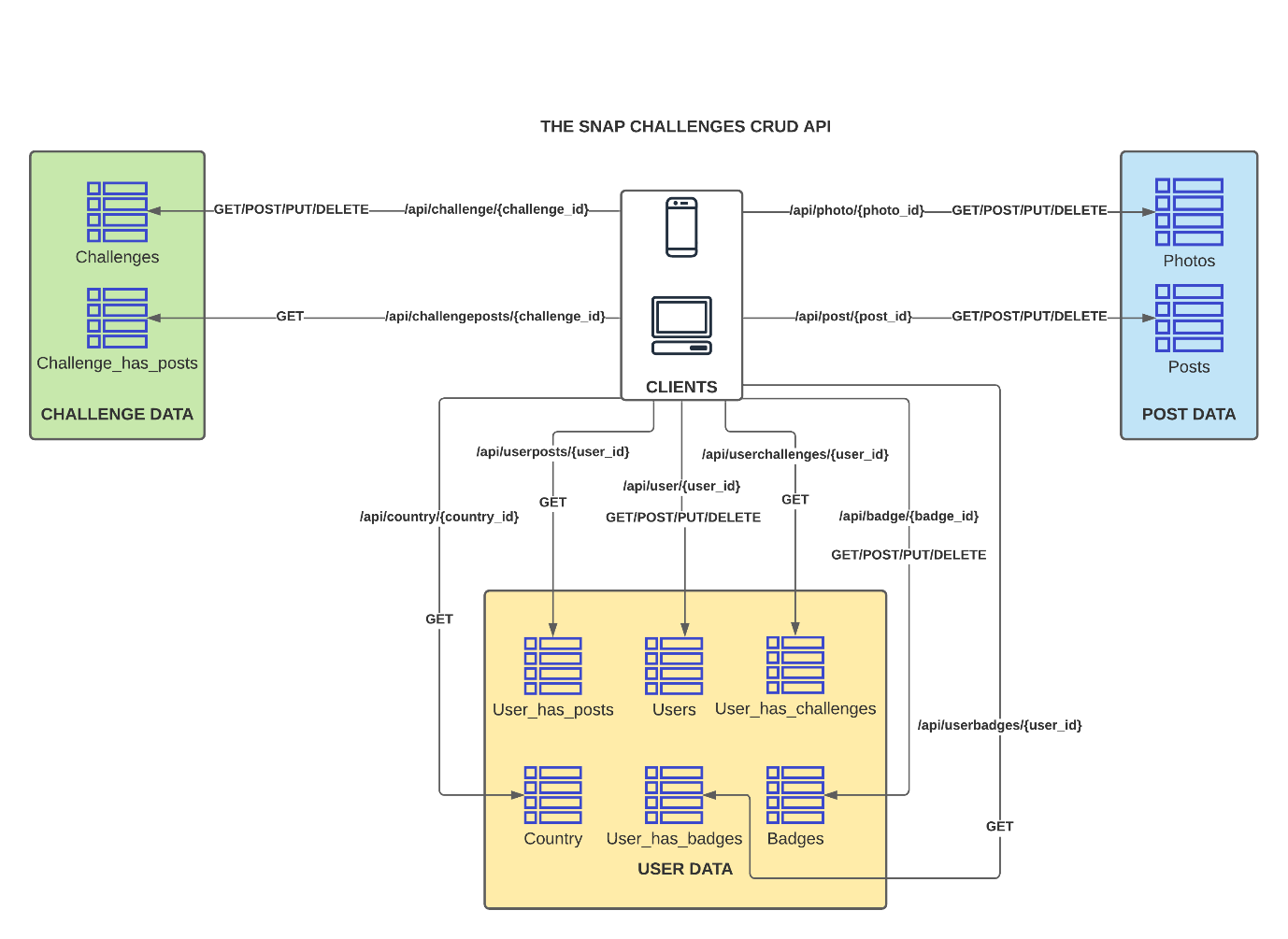
|  |  |  |
| --- | --- | --- |
| **ID** | **DESCRIPTION** | **MoSCoW** |
| FR1 | A GET request sent to the Photo API endpoint must return the json data for the specified ID as well as the 200 HTTP response code. | M |
| FR2 | A POST request sent to the Photo API endpoint must create a record of the data on the database and return the json data of that record as well as the 201 HTTP response code. | M |
| FR3 | A PUT request sent to the Photo API endpoint must update an existing record on the database and return the HTTP response code 204. | M |
| FR4 | A DELETE request sent to the User API endpoint must DELETE a record form the database if it exists and return a 204 HTTP response code. | M |
| FR5 | A GET request sent to the User API endpoint must return the json data for the specified ID as well as the 200 HTTP response code. | M |
| FR6 | A POST request sent to the User API endpoint must create a record of the data on the database and return the json data of that record as well as the 201 HTTP response code. | M |
| FR7 | A PUT request sent to the User API endpoint must update an existing record on the database and return the HTTP response code 204. | M |
| FR8 | A DELETE request sent to the User API endpoint must DELETE a record form the database if it exists and return a 204 HTTP response code. | M |
| FR9 | A GET request sent to the Country API endpoint must return the json data for the specified ID as well as the 200 HTTP response code. | M |
| FR10 | A GET request sent to the Challenge API endpoint must return the json data for the specified ID as well as the 200 HTTP response code. | M |
| FR11 | A POST request sent to the Challenge API endpoint must create a record of the data on the database and return the json data of that record as well as the 201 HTTP response code. | M |
| FR12 | A PUT request sent to the Challenge API endpoint must update an existing record on the database and return the HTTP response code 204. | M |
| FR13 | A DELETE request sent to the Challenge API endpoint must DELETE a record form the database if it exists and return a 204 HTTP response code. | M |
| FR14 | A GET request sent to the Post API endpoint must return the json data for the specified ID as well as the 200 HTTP response code. | M |
| FR15 | A POST request sent to the Post API endpoint must create a record of the data on the database and return the json data of that record as well as the 201 HTTP response code. | M |
| FR16 | A PUT request sent to the Post API endpoint must update an existing record on the database and return the HTTP response code 204. | M |
| FR17 | A DELETE request sent to the Post API endpoint must DELETE a record form the database if it exists and return a 204 HTTP response code. | M |
| FR18 | A GET request sent to the Badge API endpoint must return the json data for the specified ID as well as the 200 HTTP response code. | M |
| FR19 | A POST request sent to the Badge API endpoint must create a record of the data on the database and return the json data of that record as well as the 201 HTTP response code. | M |
| FR20 | A PUT request sent to the Badge API endpoint must update an existing record on the database and return the HTTP response code 204. | M |
| FR21 | A DELETE request sent to the Badge API endpoint must DELETE a record form the database if it exists and return a 204 HTTP response code. | M |
| FR22 | A GET request sent to the ChallengePosts API endpoint must return the json data for the specified ID as well as the 200 HTTP response code. | M |
| FR23 | A GET request sent to the UserPosts API endpoint must return the json data for the specified ID as well as the 200 HTTP response code. | M |
| FR24 | A GET request sent to the UserChallenges API endpoint must return the json data for the specified ID as well as the 200 HTTP response code. | M |
| FR25 | A GET request sent to the UserBadges API endpoint must return the json data for the specified ID as well as the 200 HTTP response code. | M |
| FR26 | Certain information may only be retrieved from the API if an authorization token is provided alongside the initial request. | M |

# **4 – Methodology**

# **5 – Design**

****

****

****

# **6 – Implementation**

# **7 – Evaluation**

# **8 – Conclusion**

# **9 – References**

# **10 – Appendices**